图像缩放与补全

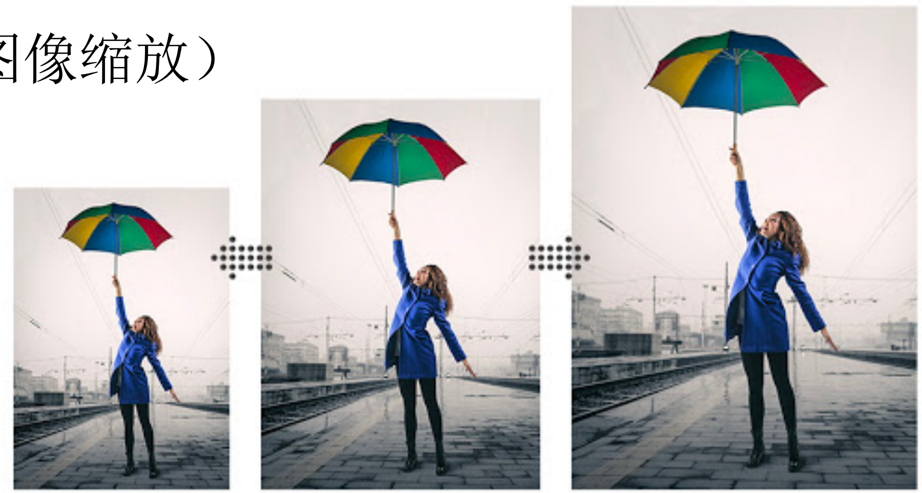
周晓巍

www.cad.zju.edu.cn/home/xzhou/

Some slides borrowed from Guofeng Zhang (ZJU), Lingqi Yan (UCSB)

Outline

- Image resizing （图像缩放）



- Image completion （图像补全）

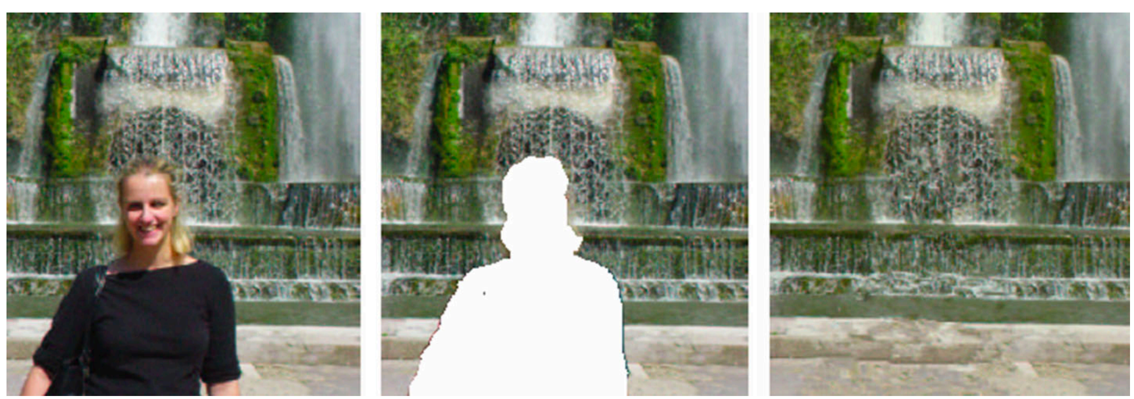
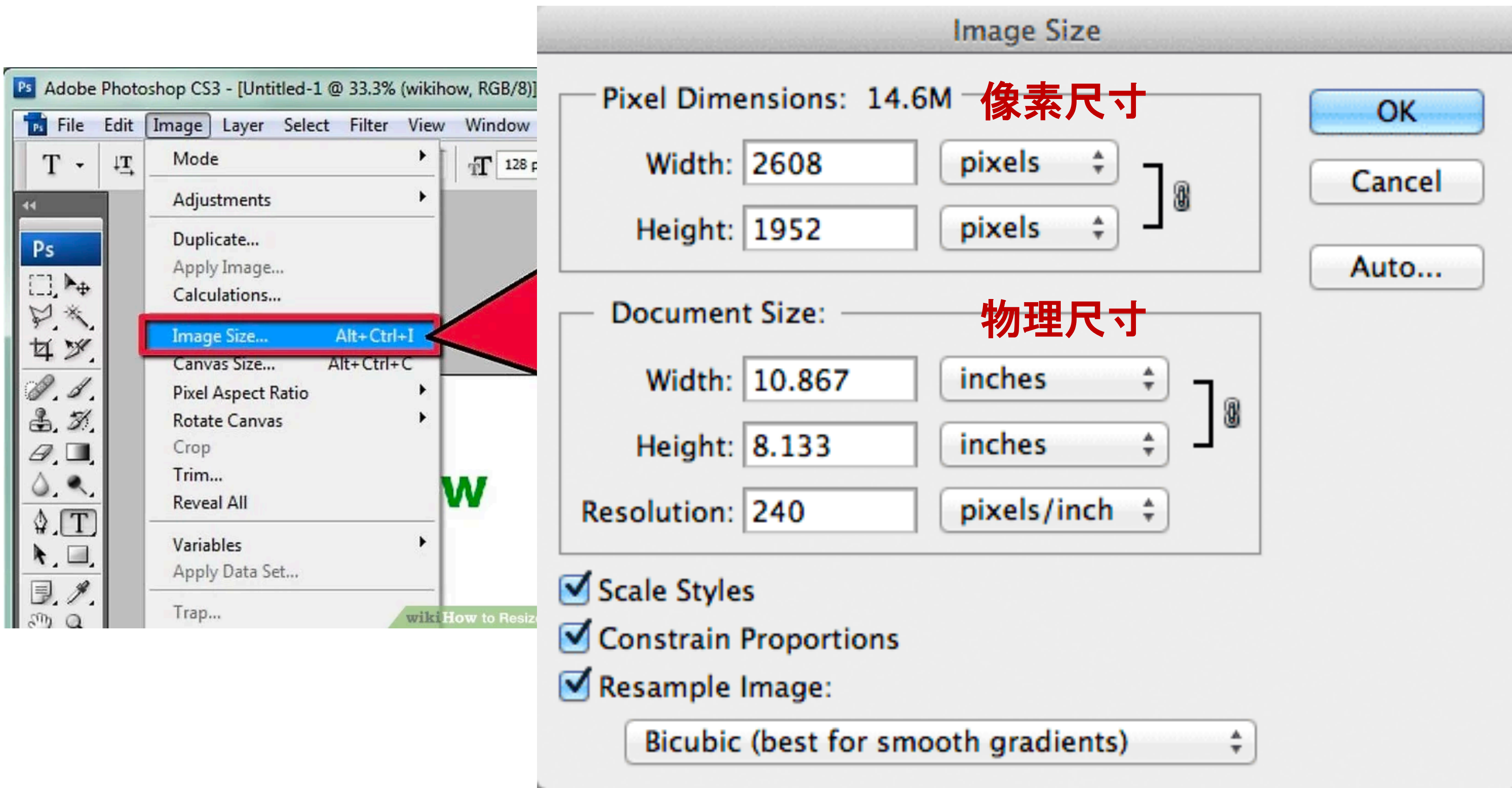


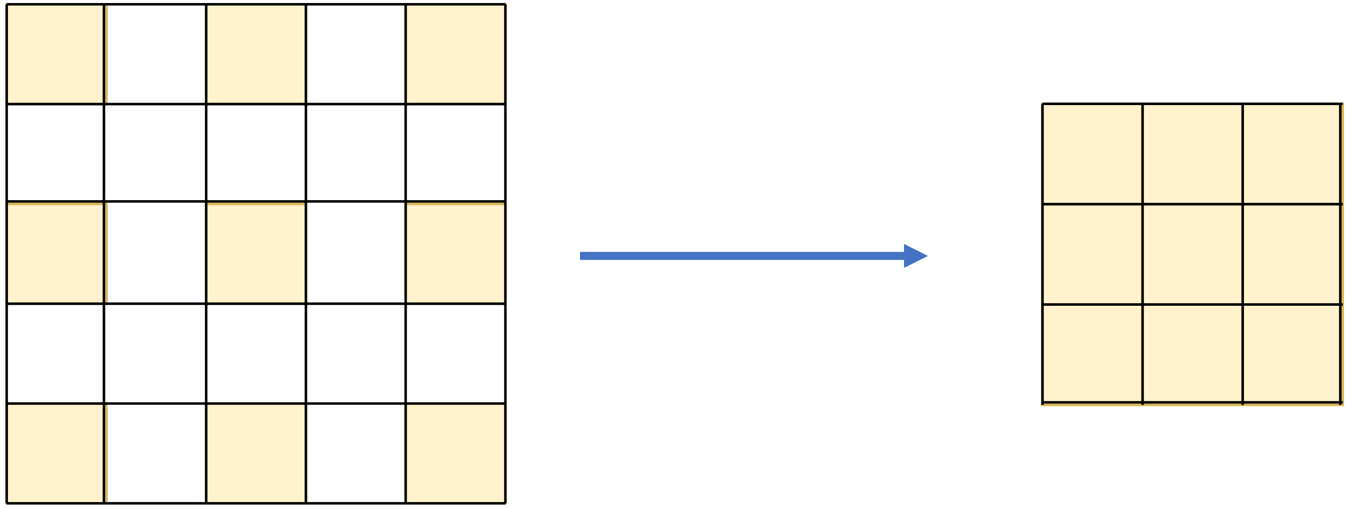
Image resizing

Change image size / resolution in Photoshop



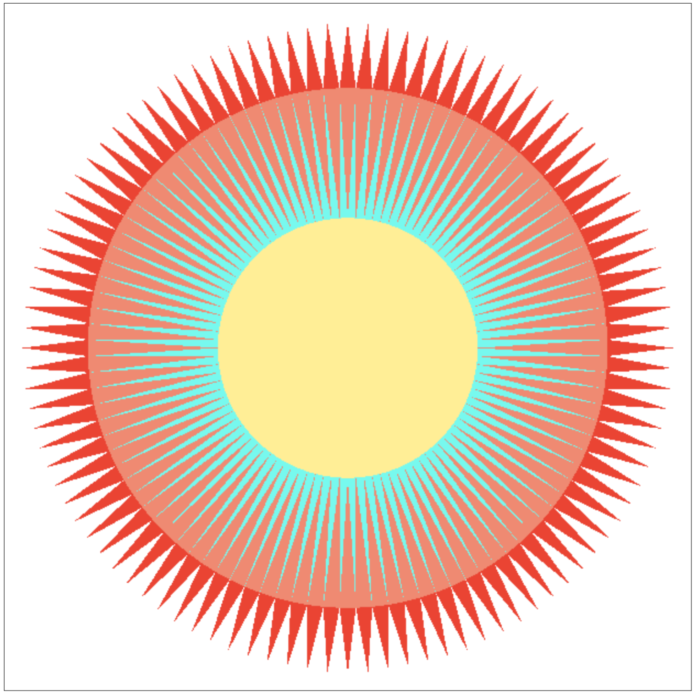
Sampling

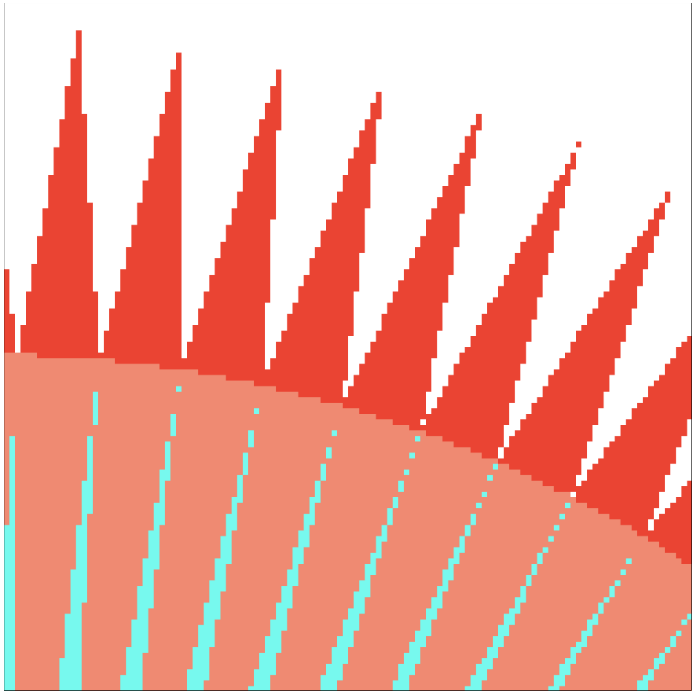
Reducing image size – down-sampling



Is sampling really so easy?

Jaggies (Staircase Pattern)

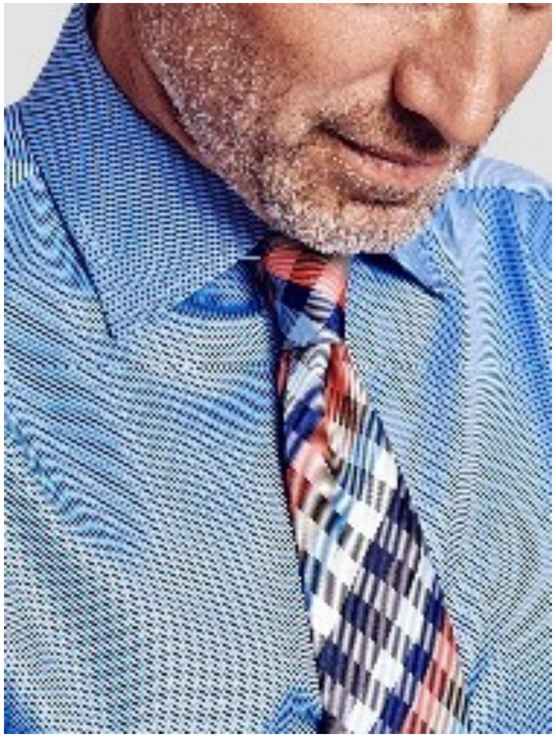




Is sampling really so easy?

Moiré Patterns in Imaging

[mwa:]



lystit.com

Skip odd rows and columns

Is sampling really so easy?

Wagon Wheel Illusion (False Motion)

Aliasing（走样）

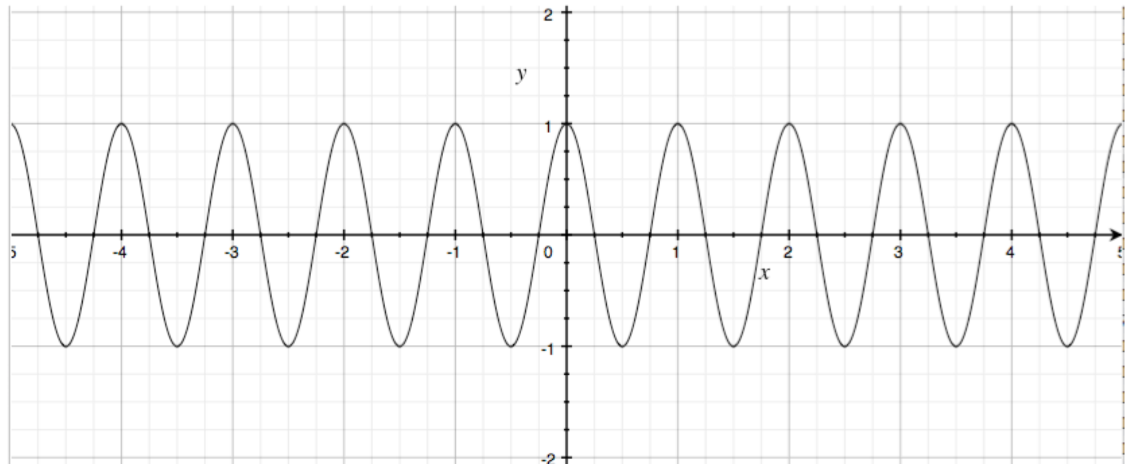
Aliasing - artifacts due to sampling

- Jaggies / Moire effect – undersampling in space
- Wagon wheel effect – undersampling in time

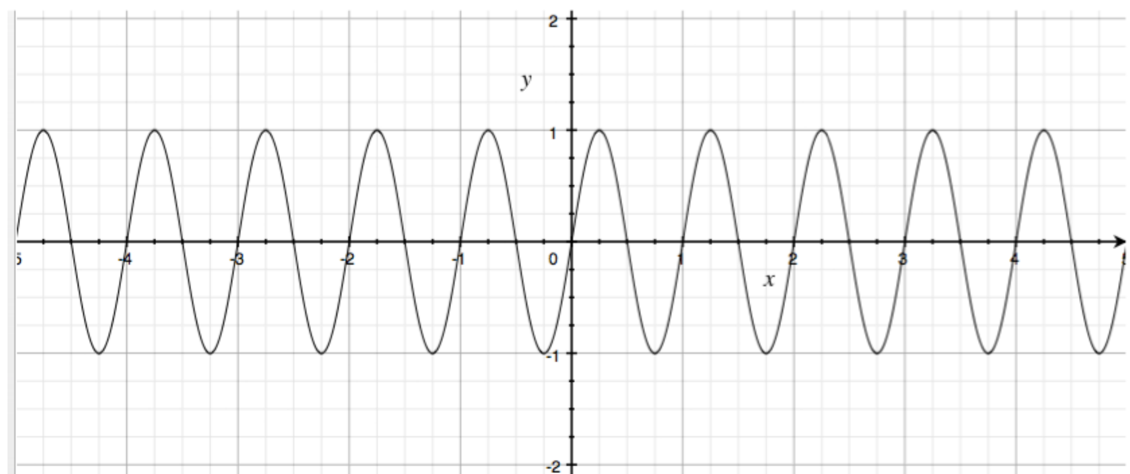
Why does aliasing happen?

- Signals are changing too fast (high frequency),
  but sampled too slow

Sines and Cosines



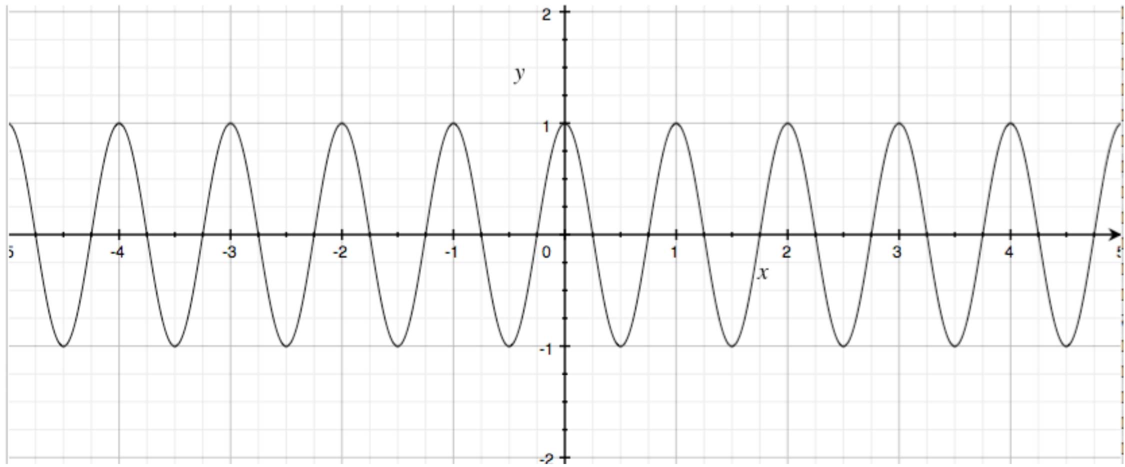
$$\cos 2\pi x$$



$$\sin 2\pi x$$

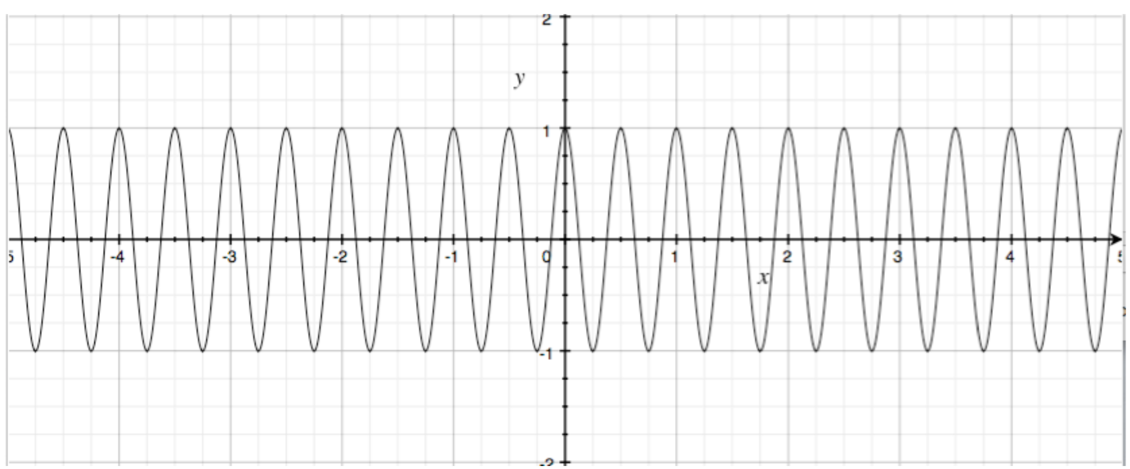
Frequencies $\cos 2\pi f x$

$$f = \frac{1}{T}$$



$\cos 2\pi x$

$$f = 1$$



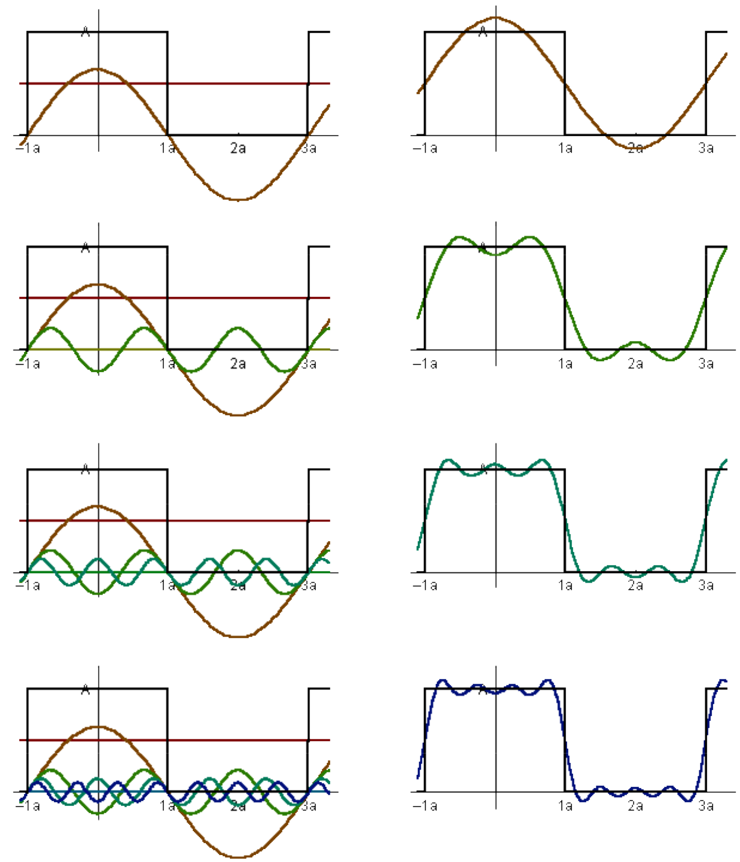
$$f = 2$$

$\cos 4\pi x$

Fourier Transform

Represent a function as a weighted sum of sines and cosines

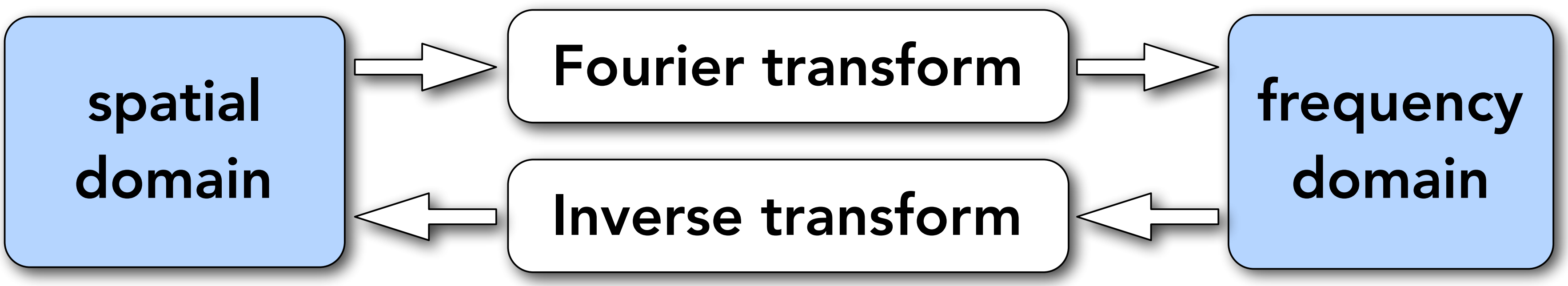
Joseph Fourier 1768 - 1830



$$f(x) = \frac{A}{2} + \frac{2A\cos(t\omega)}{\pi} - \frac{2A\cos(3t\omega)}{3\pi} + \frac{2A\cos(5t\omega)}{5\pi} - \frac{2A\cos(7t\omega)}{7\pi} + \cdots$$

Fourier Transform Decomposes A Signal Into Frequencies

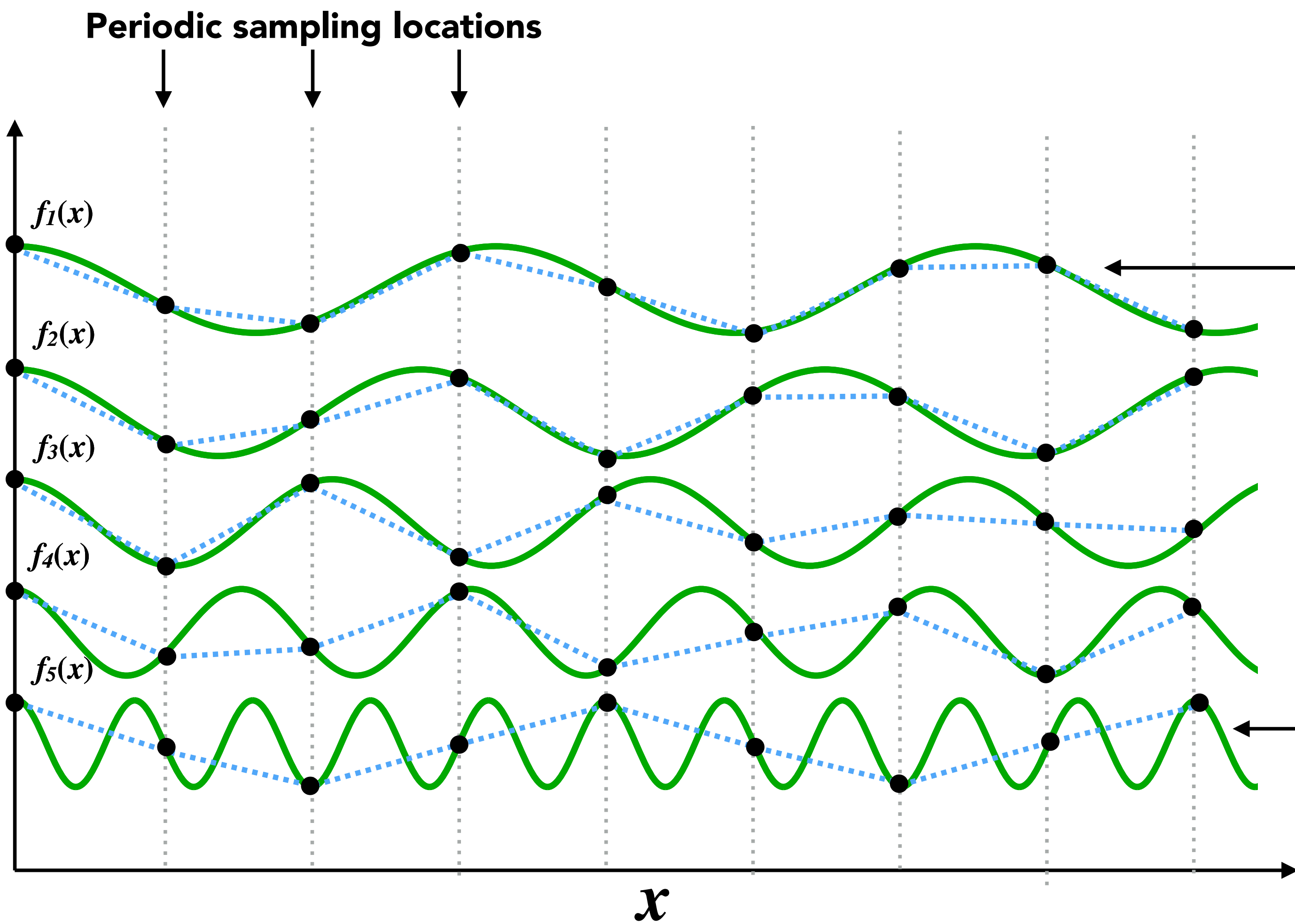
$$f(x) \qquad F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \omega x} dx \qquad F(\omega)$$



$$f(x) = \int_{-\infty}^{\infty} F(\omega) e^{2\pi i \omega x} d\omega$$

Recall $e^{ix} = \cos x + i \sin x$

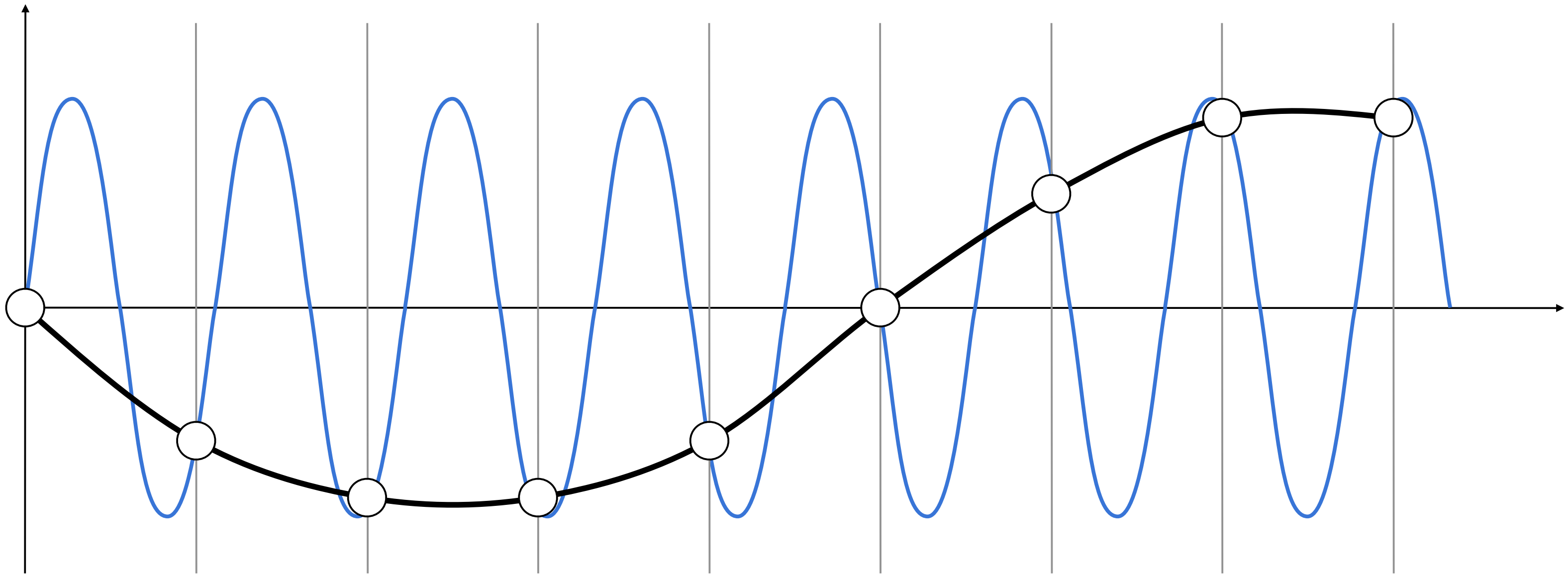
Higher Frequencies Need Faster Sampling



Low-frequency signal: sampled adequately for reasonable reconstruction

High-frequency signal is insufficiently sampled: reconstruction incorrectly appears to be from a low frequency signal

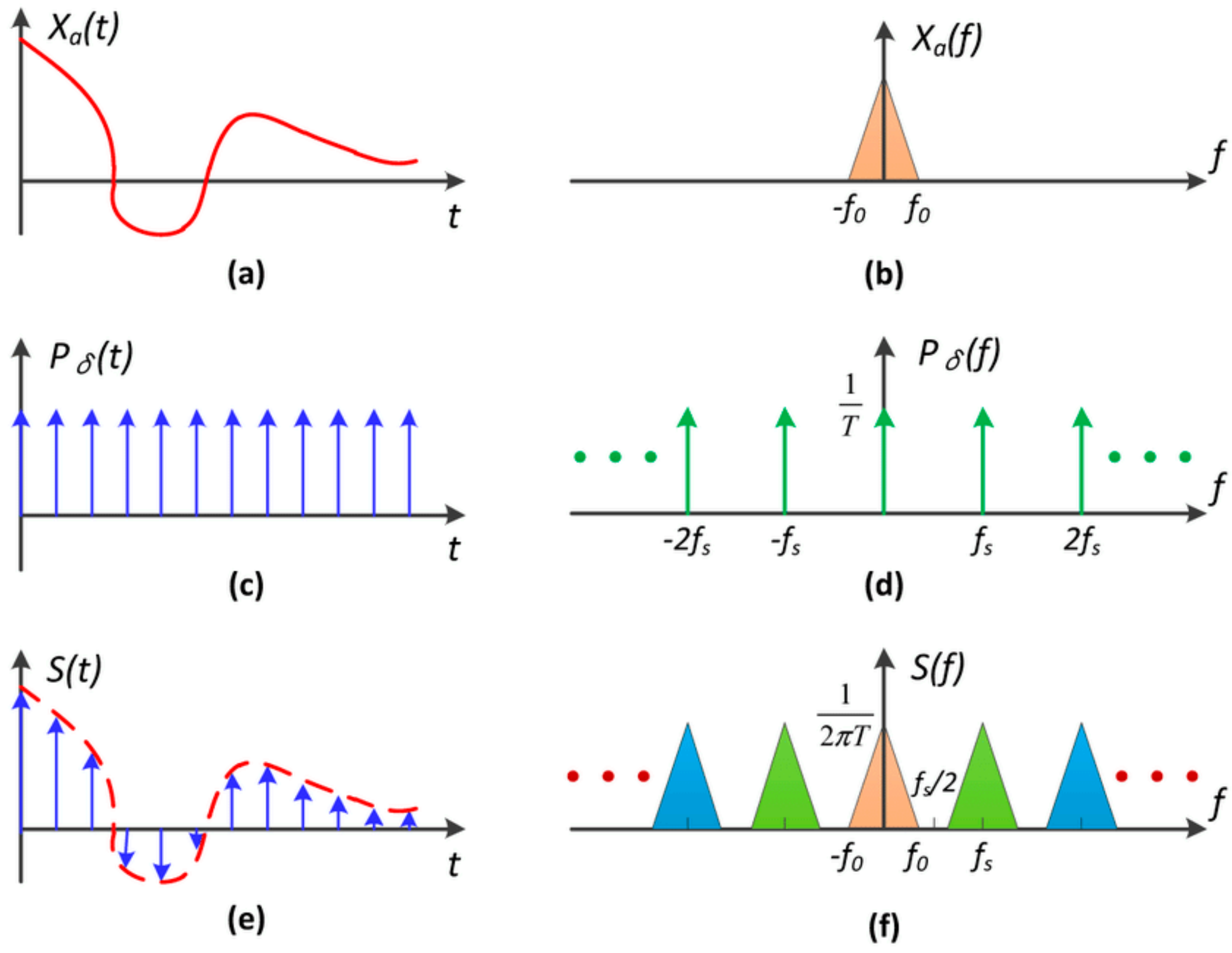
Undersampling Creates Frequency Aliases



High-frequency signal is insufficiently sampled: samples erroneously appear to be from a low-frequency signal

Two frequencies that are indistinguishable at a given sampling rate are called “aliases”

Sampling = Repeating Frequency Contents

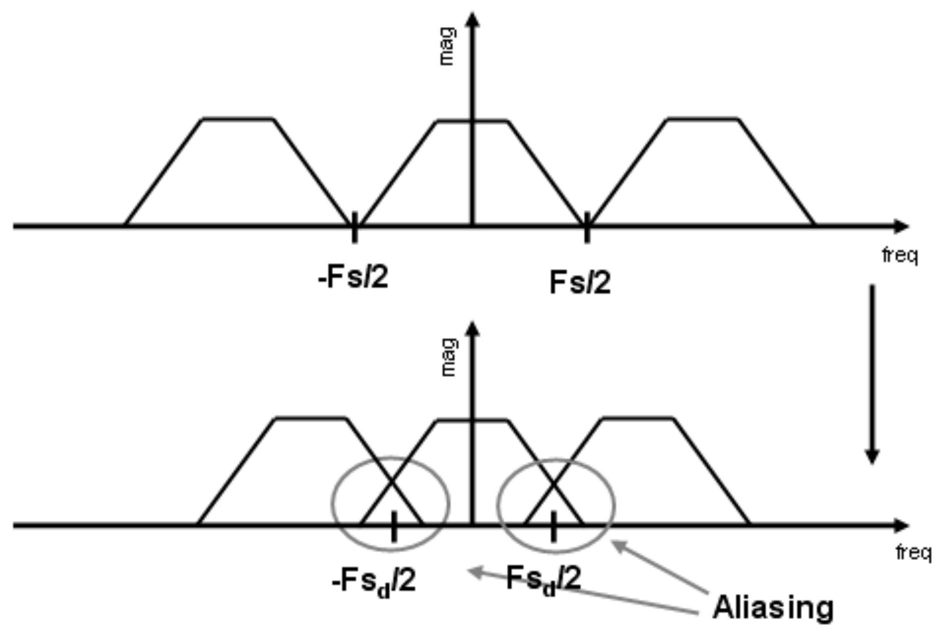


https://www.researchgate.net/figure/The-evolution-of-sampling-theorem-a-The-time-domain-of-the-band-limited-signal-and-b_fig5_301556095

Aliasing = Mixed Frequency Contents

Dense sampling:

Sparse sampling:



How can we reduce aliasing?

Option 1: Increasing sampling rate

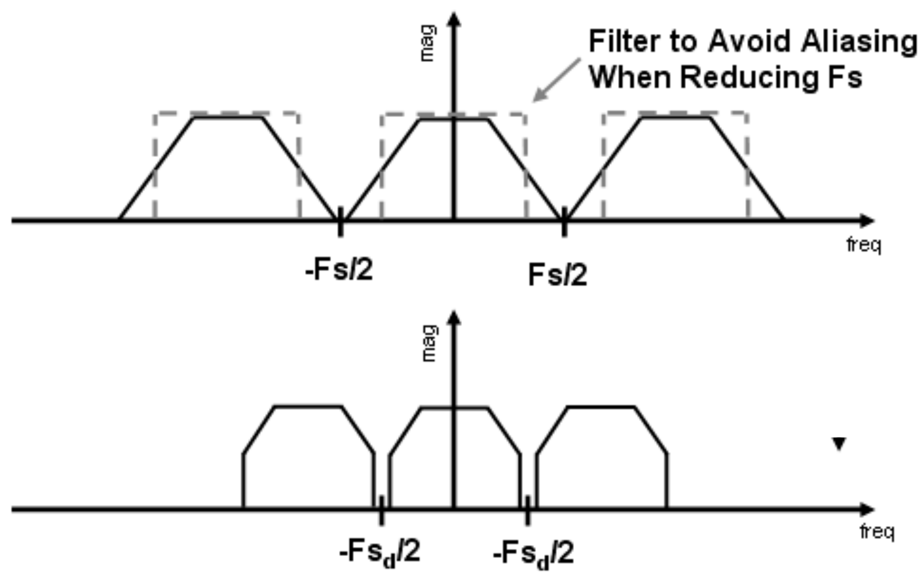
Option 2: Anti-aliasing

Filtering out high frequendcies before sampling

Antialiasing = Limiting, then repeating

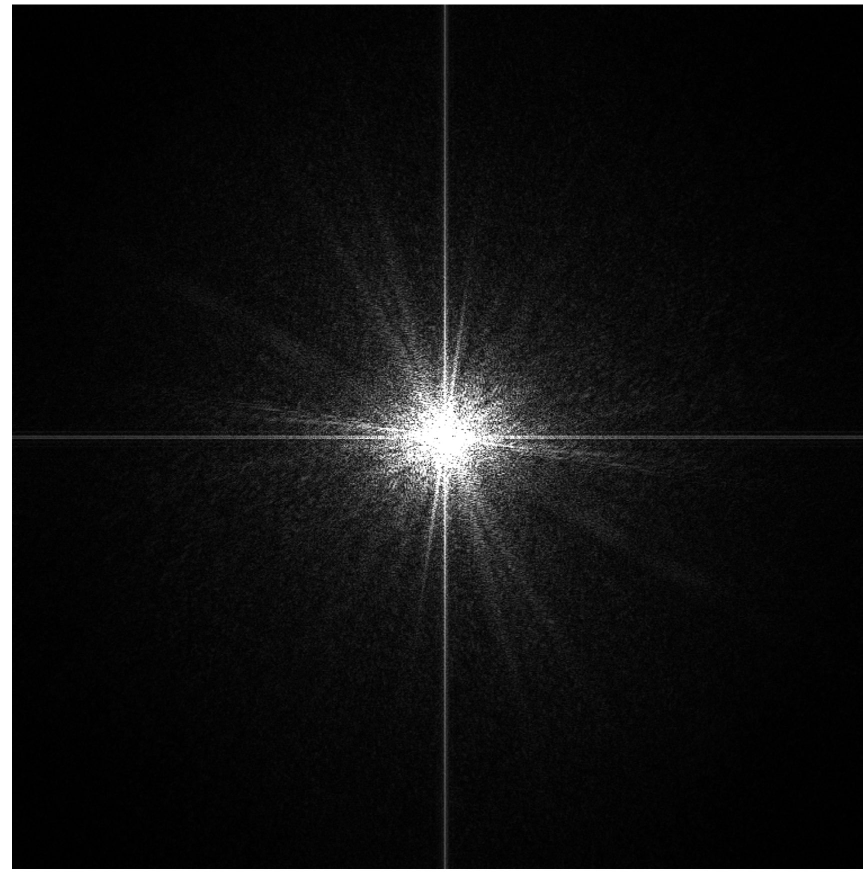
Filtering

Then sparse sampling

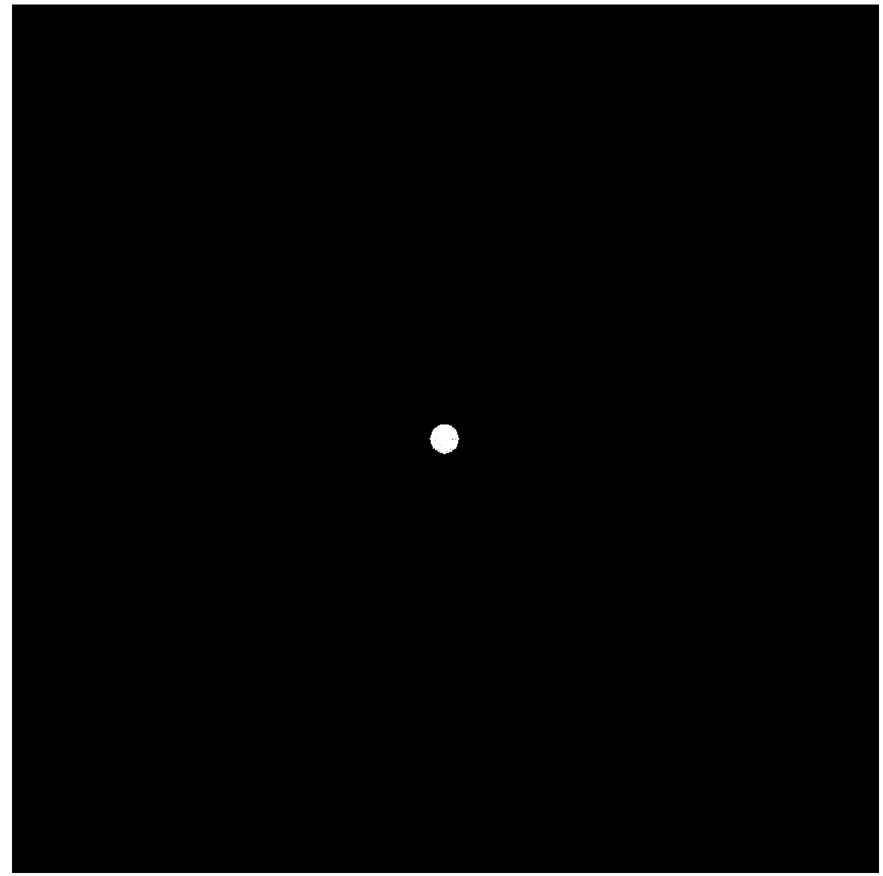


Filtering = Getting rid of certain frequency contents

Visualizing Image Frequency Content

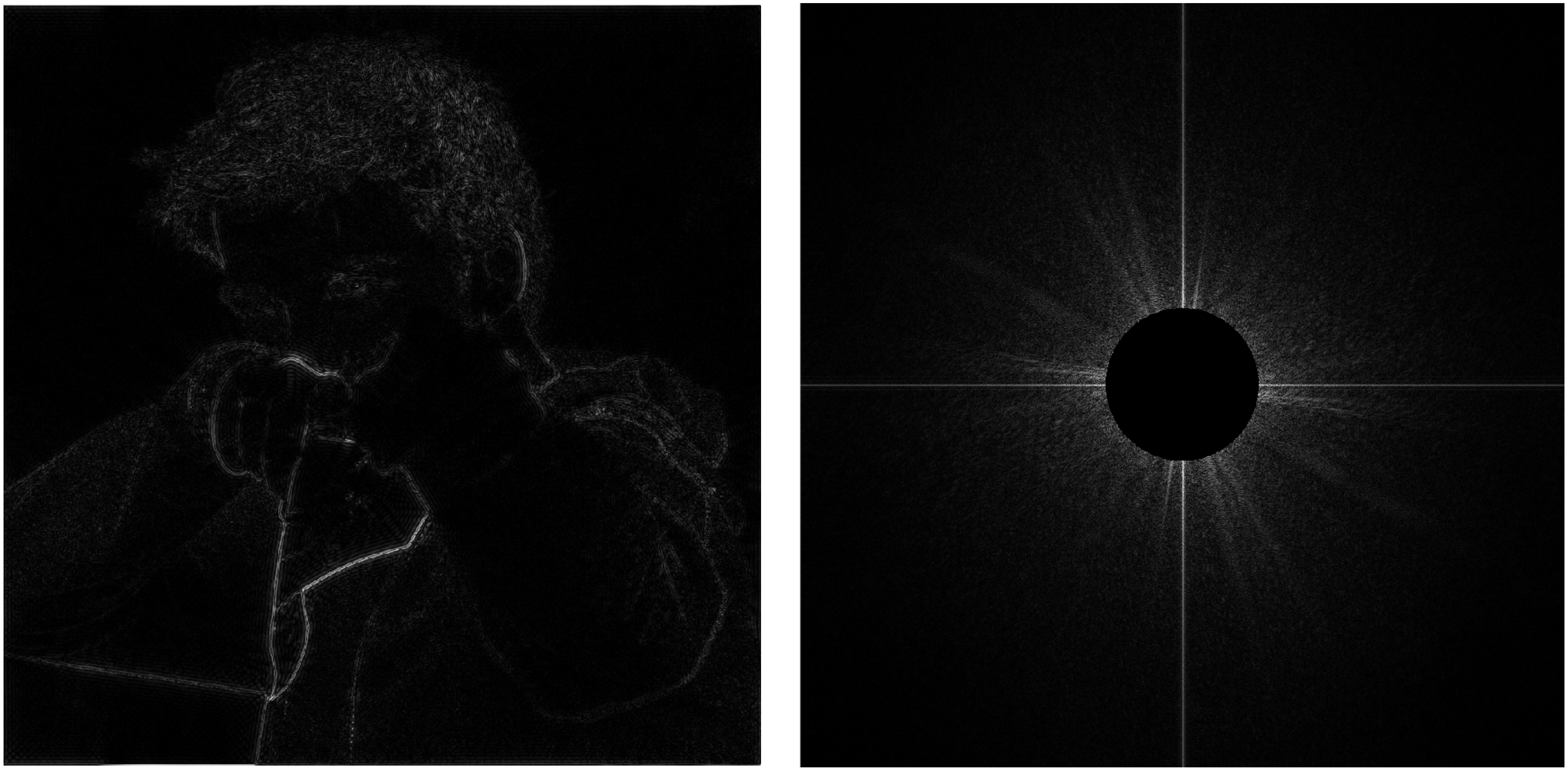


Filter Out High Frequencies (Blur)



Low-pass filter

Filter Out Low Frequencies Only (Edges)



High-pass filter

Filtering = Convolution
(= Averaging)

Convolution

Signal	1	3	5	3	7	1	3	8	6	4

Filter	1/4	1/2	1/4

Point-wise local averaging in a “sliding window”

Convolution

Signal

1	3	5	3	7	1	3	8	6	4

Filter

1/4	1/2	1/4

$$1 \times (1/4) + 3 \times (1/2) + 5 \times (1/4) = 3$$

Result

	3								

Convolution

Signal

1	3	5	3	7	1	3	8	6	4

Filter

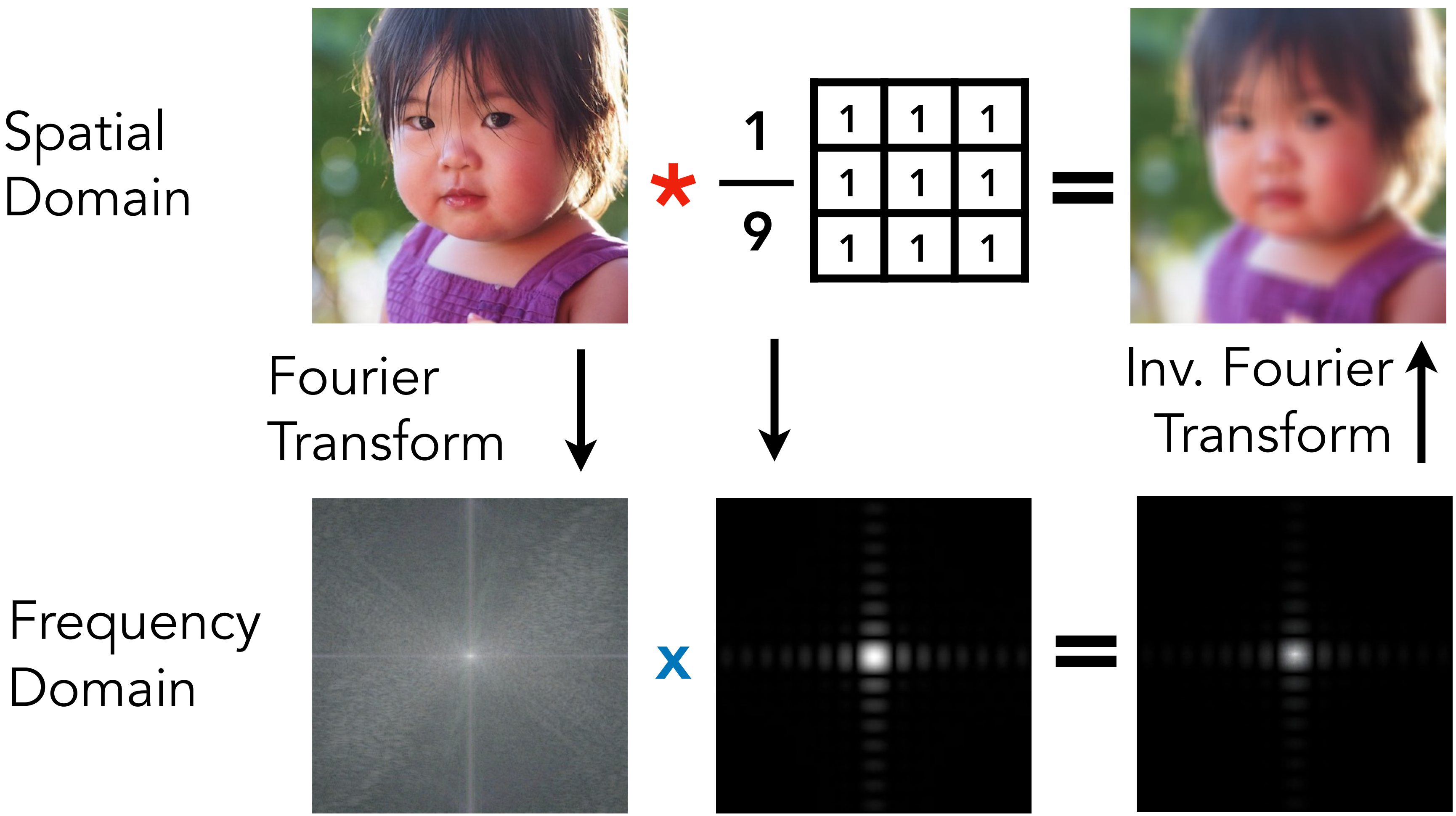
1/4	1/2	1/4

$$3 \times (1/4) + 5 \times (1/2) + 3 \times (1/4) = 4$$

Result

	3	4							

Convolution Theorem



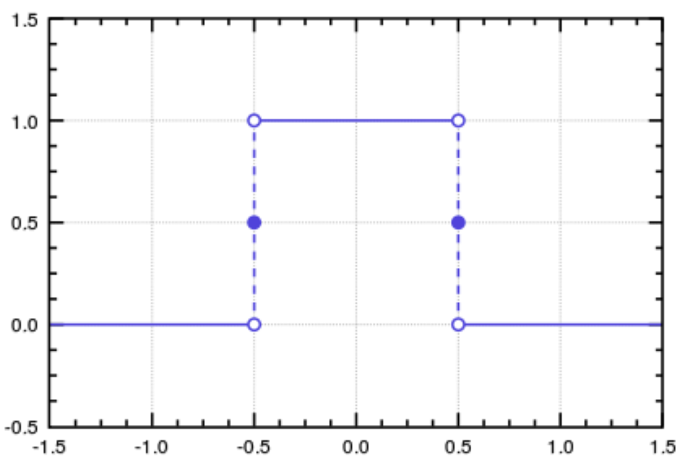
Box Filter

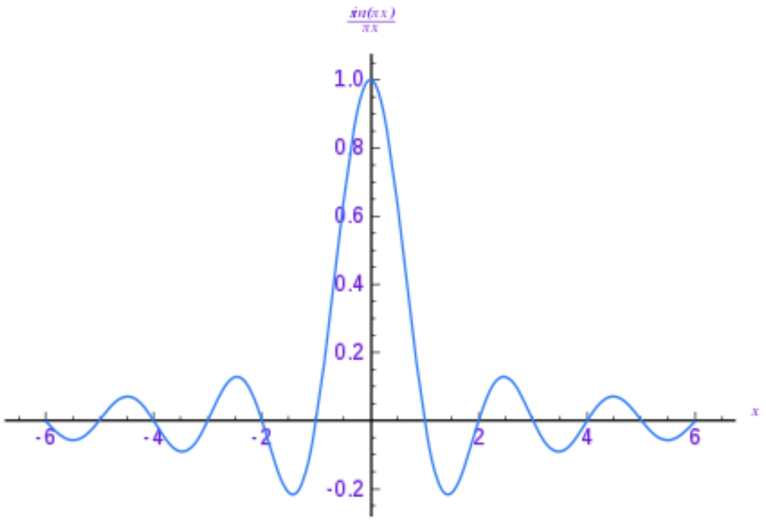
$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Example: 3x3 box filter

Box Filter

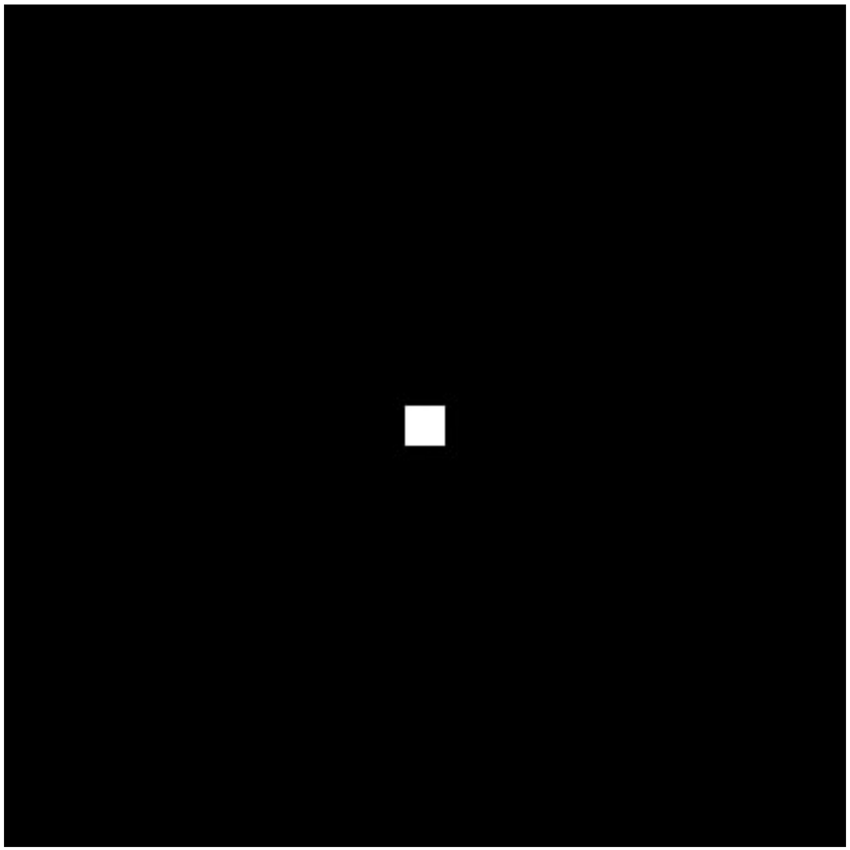
What is the Fourier transform of a rectangular function?

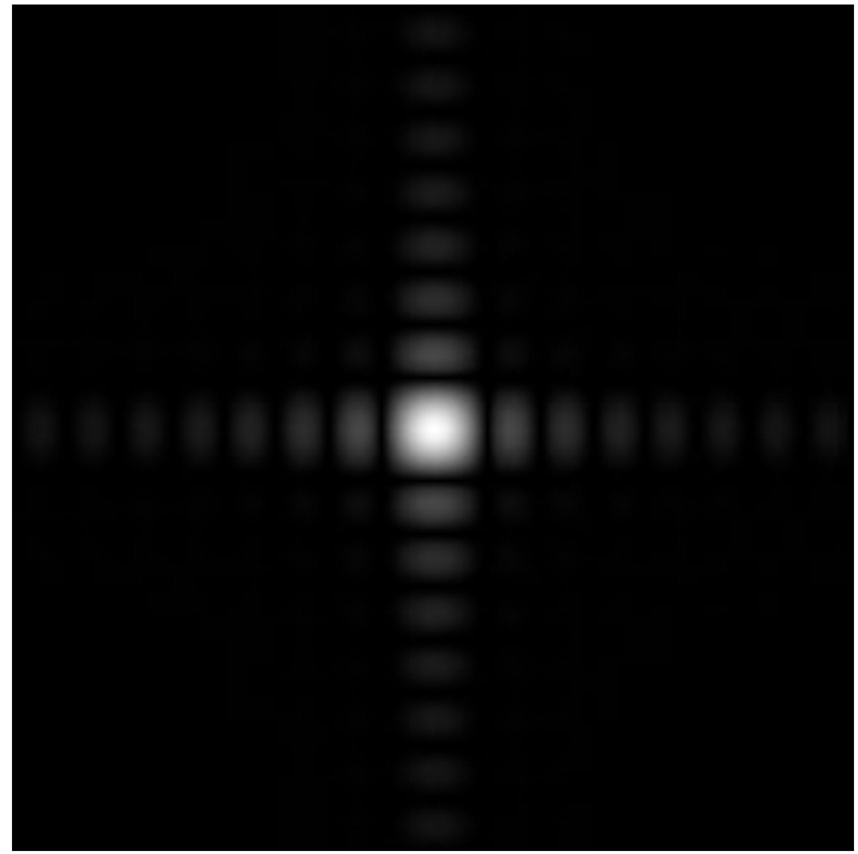




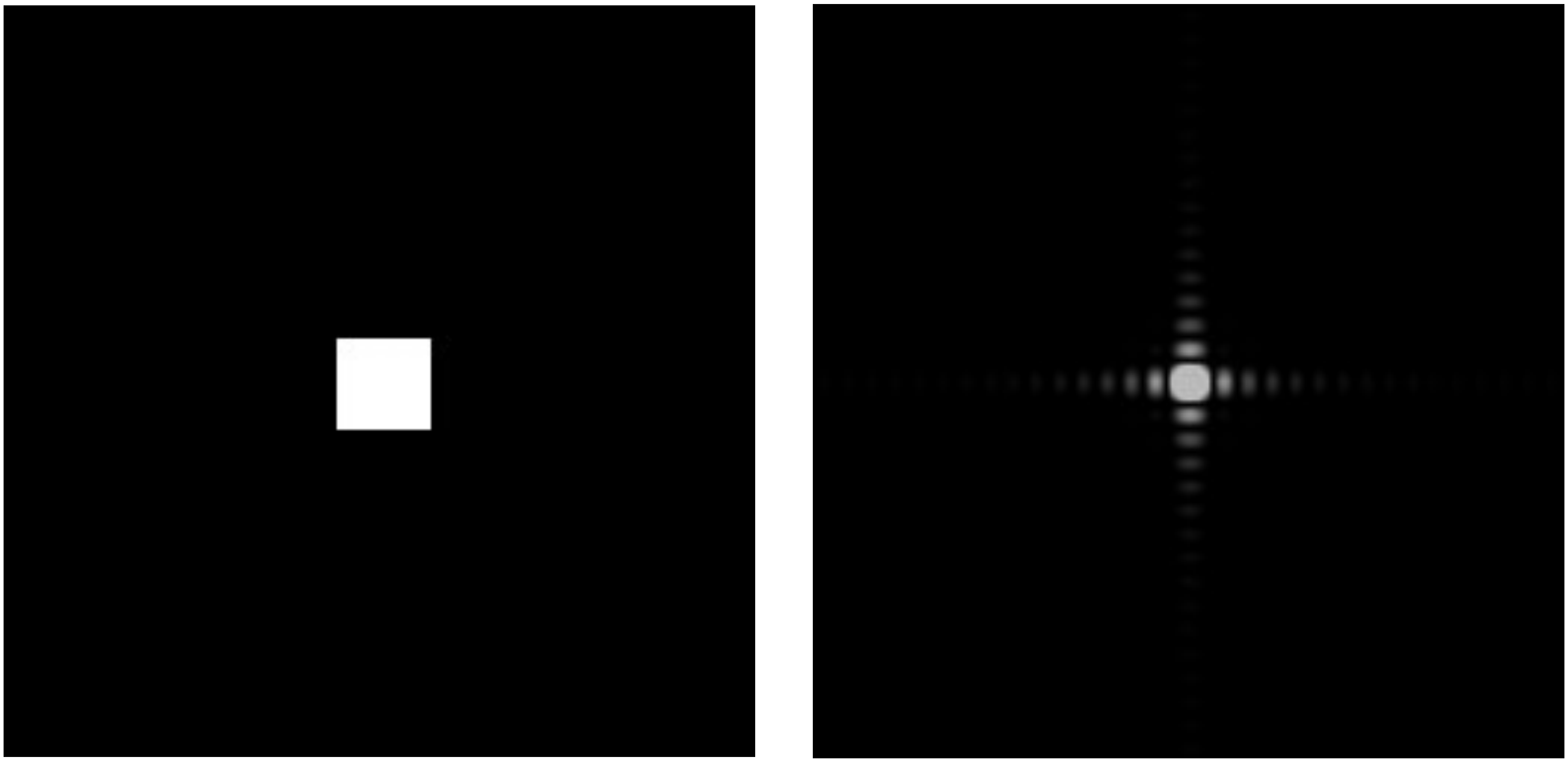
$$\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}.$$

Box Function = "Low Pass" Filter

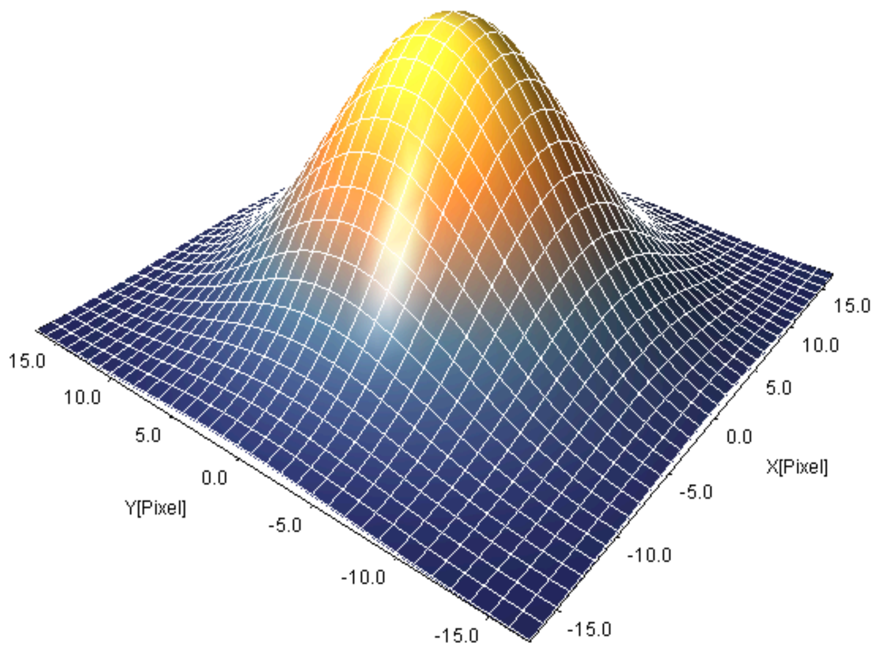




Wider Filter Kernel = Lower Frequencies



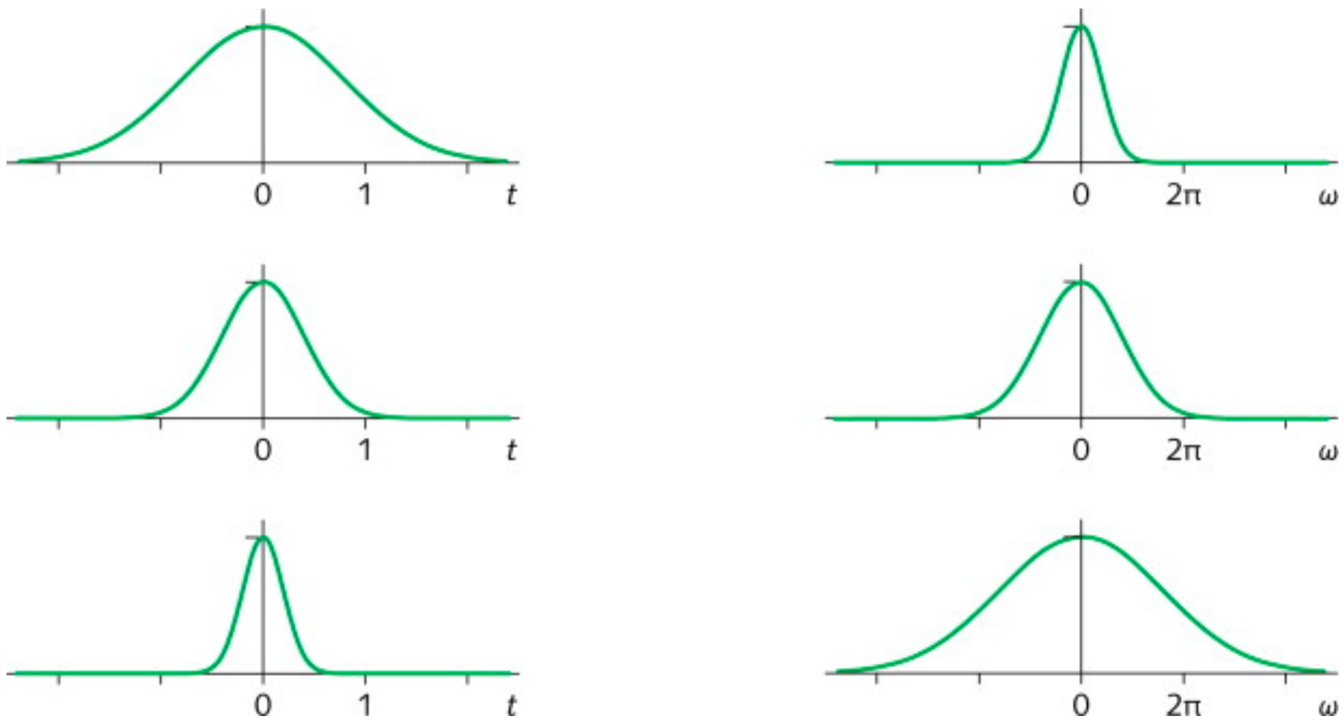
Gaussian filter



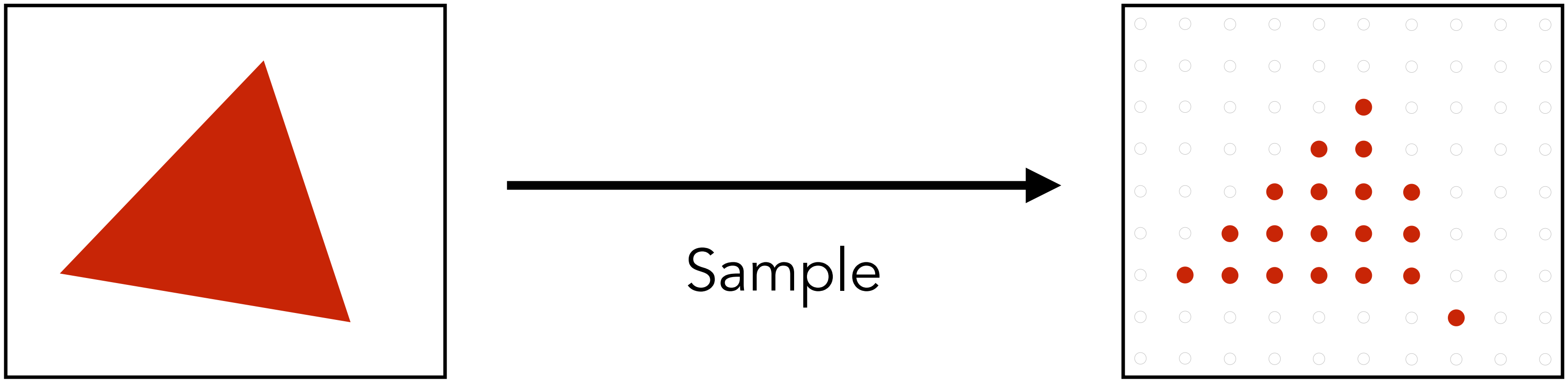
$$f(x,y) = A \exp\left(- \left(\frac{(x-x_o)^2}{2\sigma_X^2} + \frac{(y-y_o)^2}{2\sigma_Y^2}\right)\right).$$

Guanssian filter

What is the Fourier transform of a Gaussian?

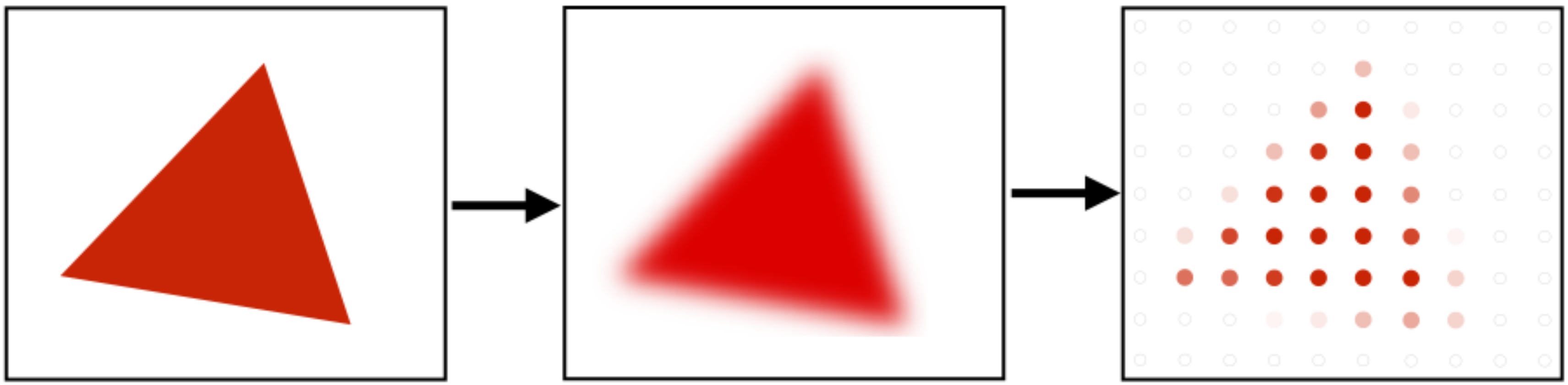


Regular Sampling



Note jaggies in rasterized triangle
where pixel values are pure red or white

Antialiased Sampling



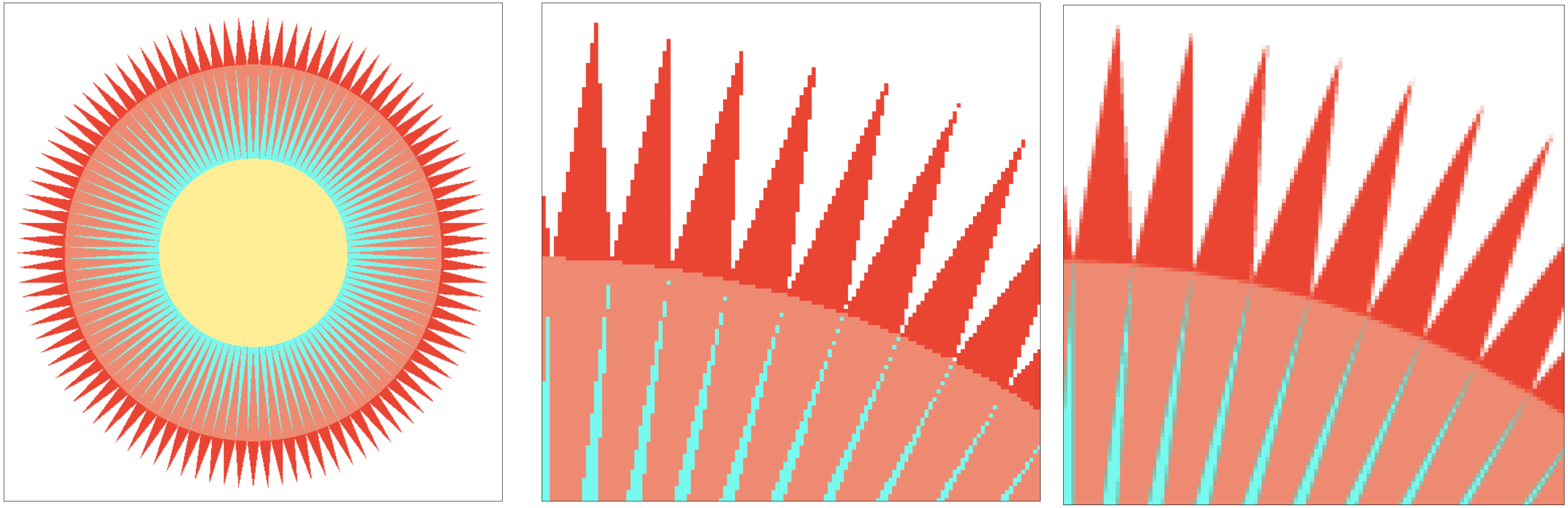
Pre-Filter

(remove frequencies above Nyquist)

Sample

Note antialiased edges in rasterized triangle
where pixel values take intermediate values

Antialiasing



Antialiasing

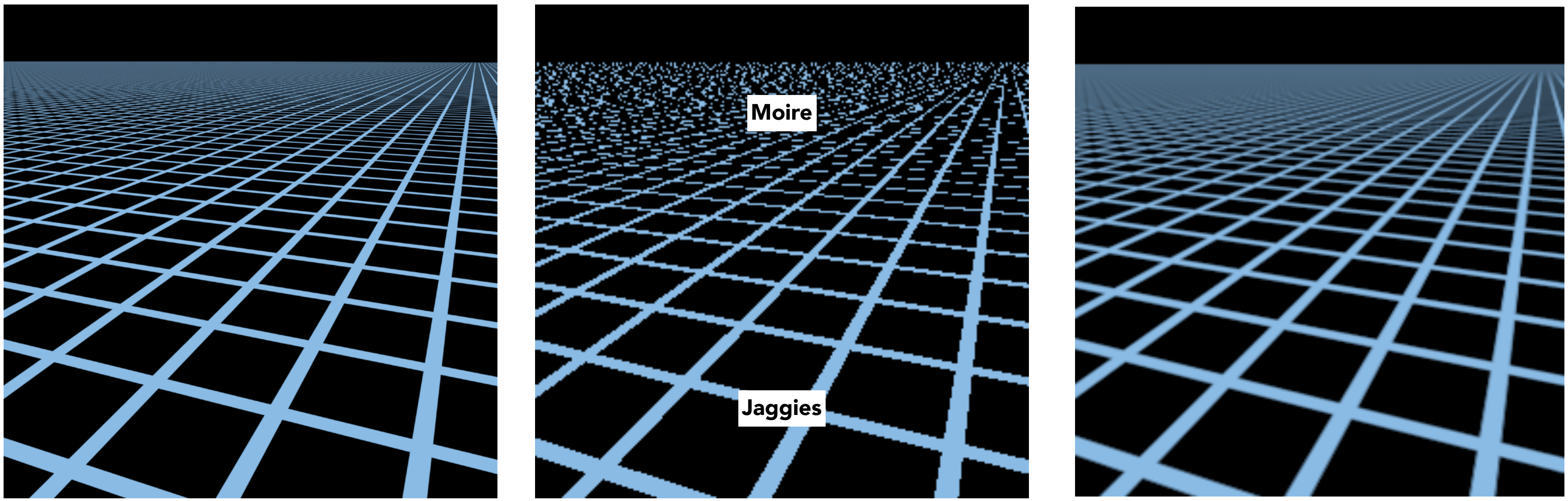
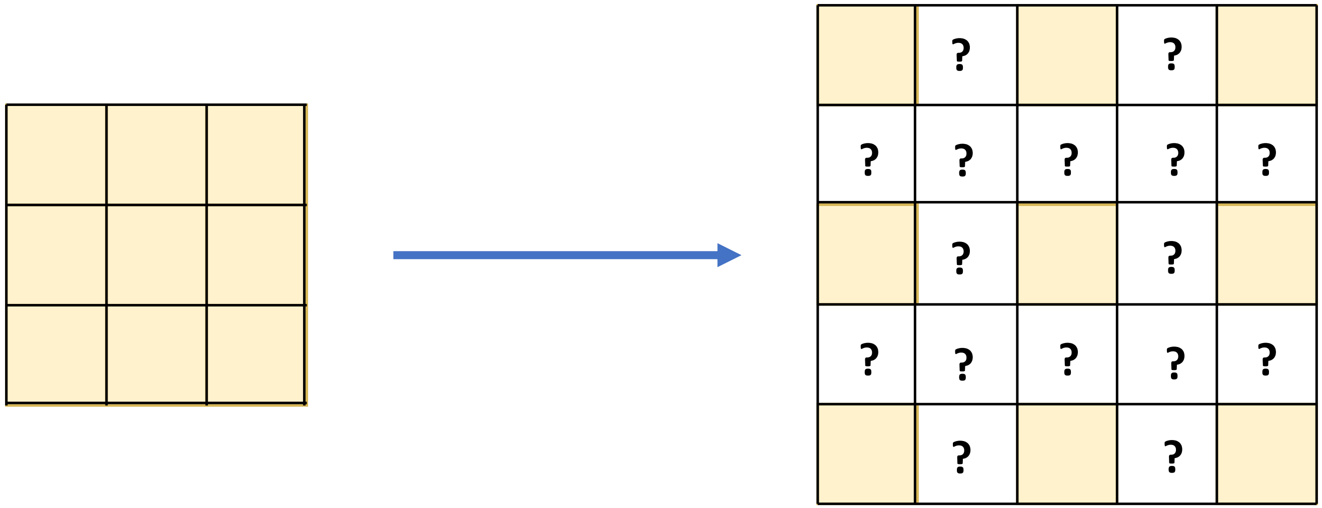


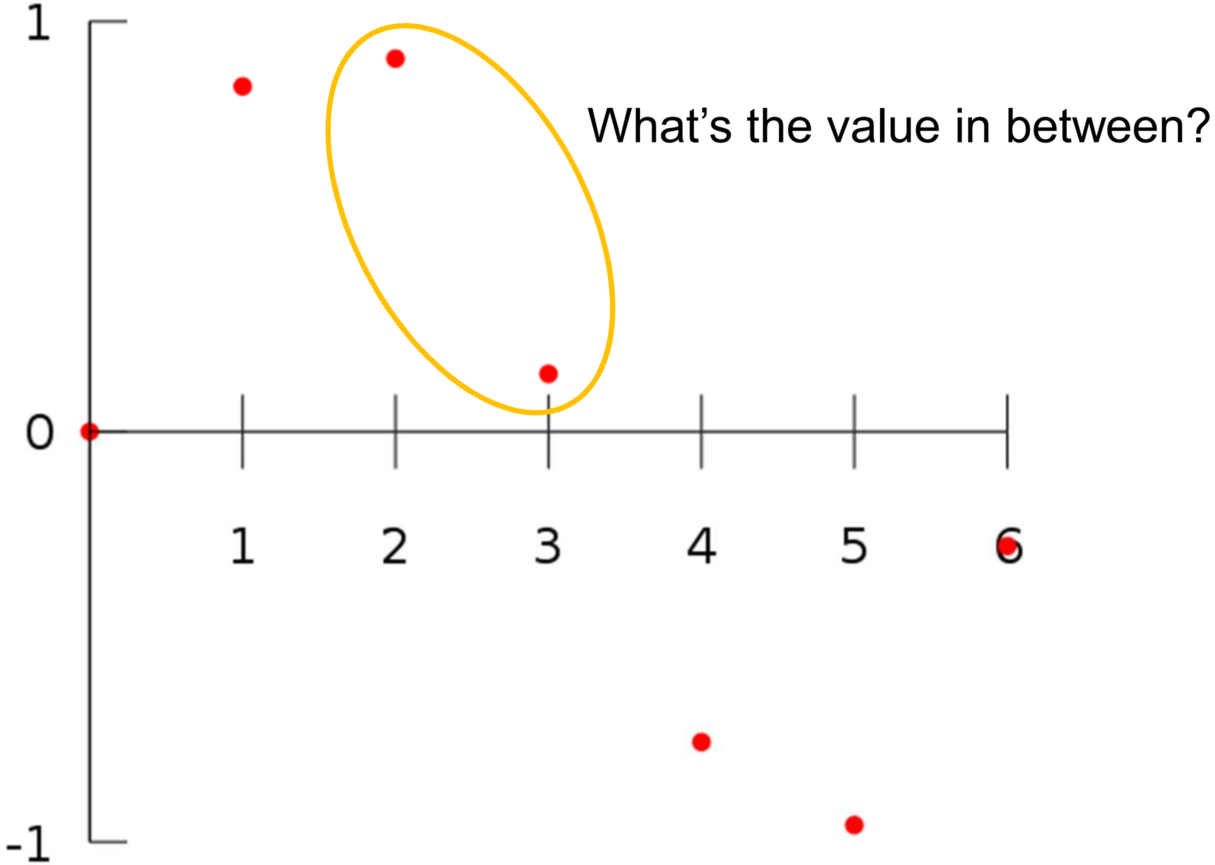
Image magnification

Image magnification

Inverse of down-sampling (up-sampling)



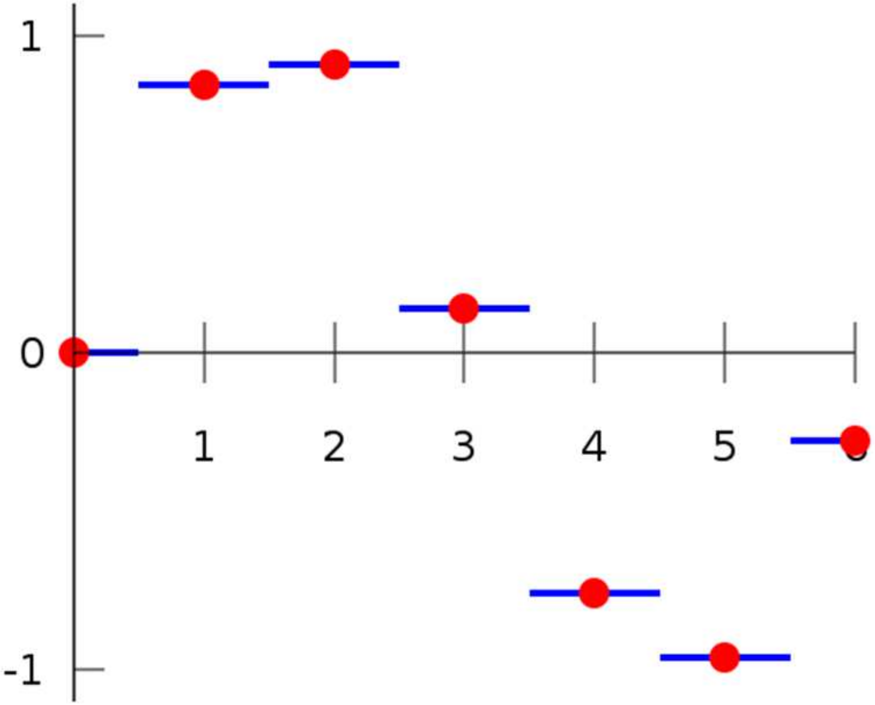
Interpolation



Nearest-neighbor interpolation

Not continuous

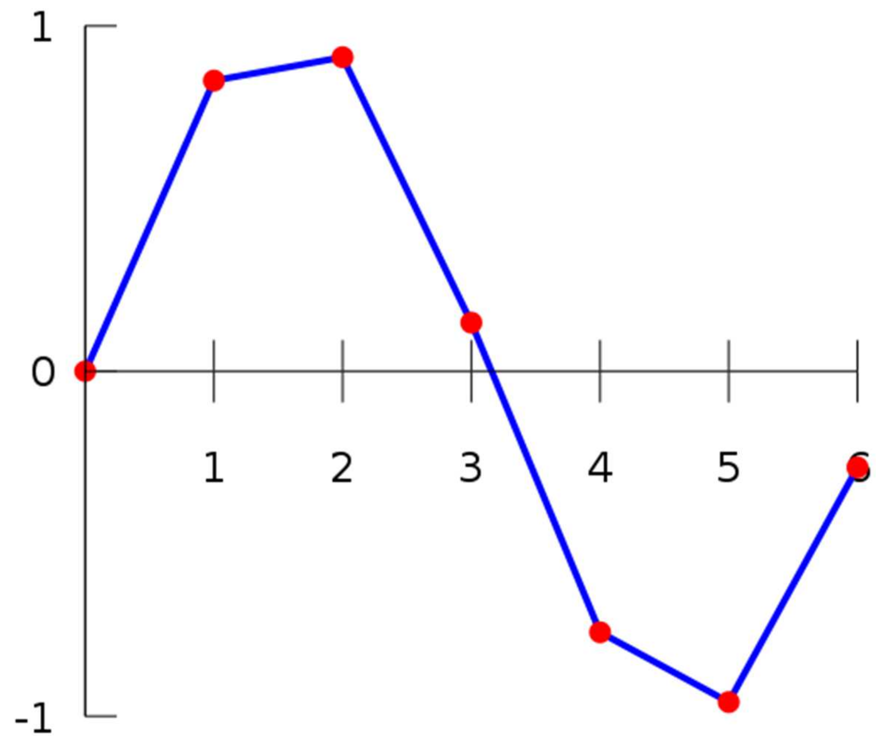
Not smooth



Linear interpolation

Continous

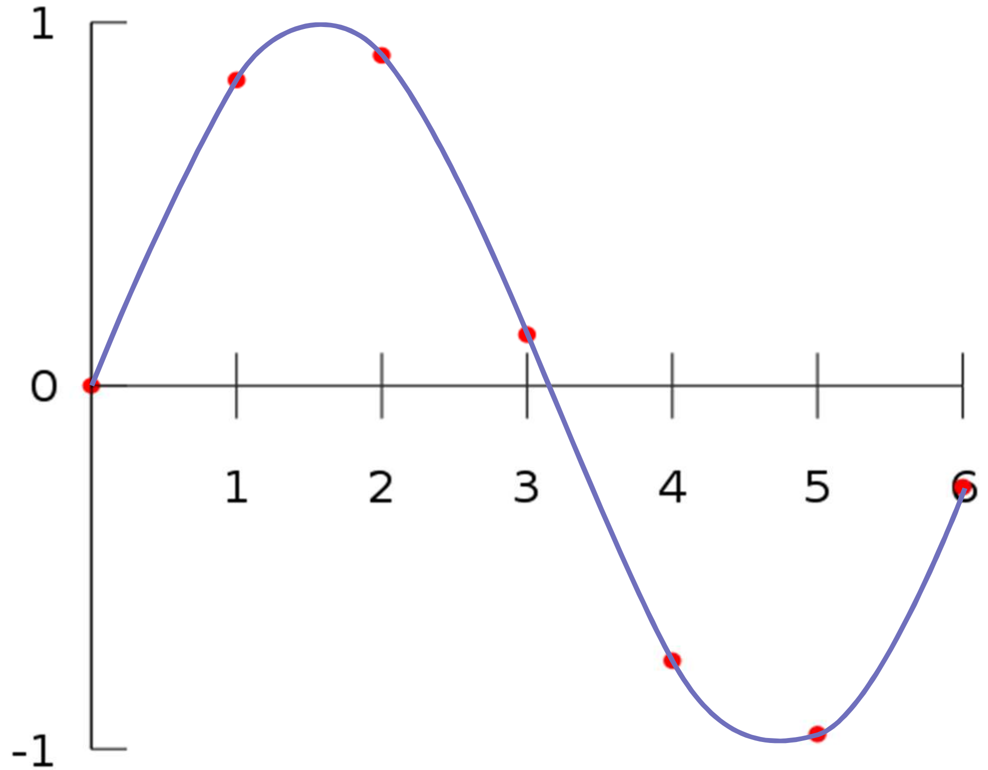
Not smooth



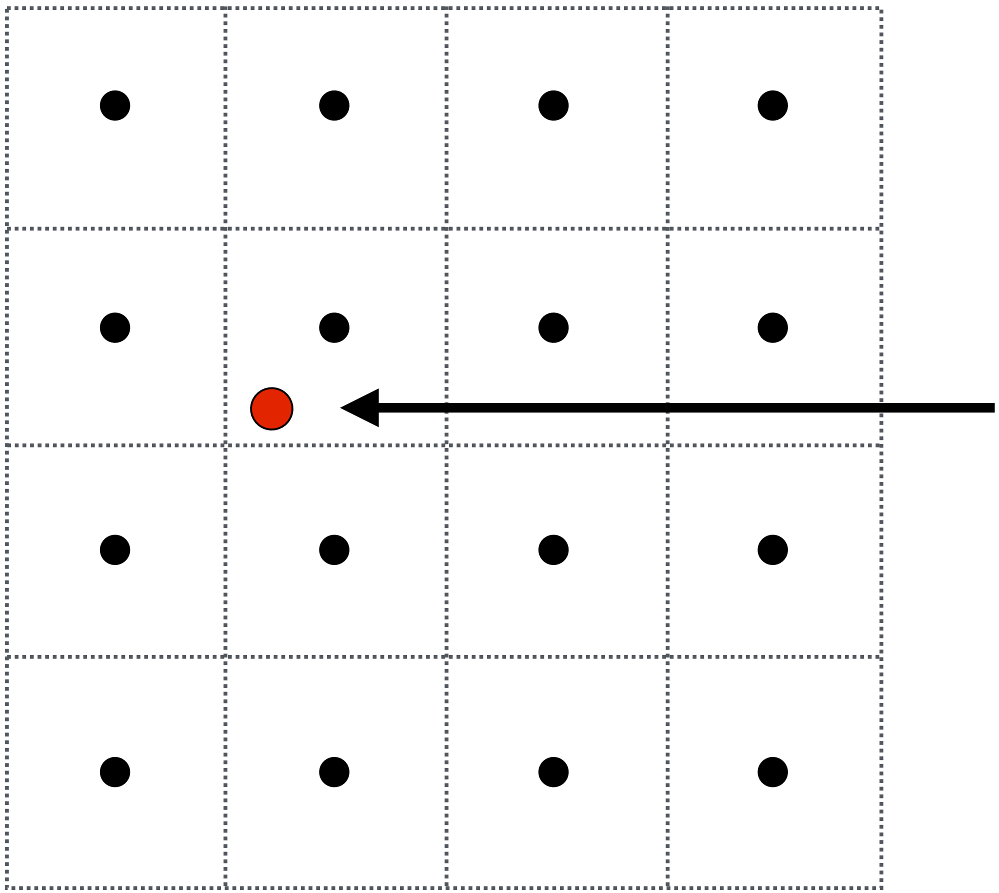
Cubic interpolation

Continous

Smooth



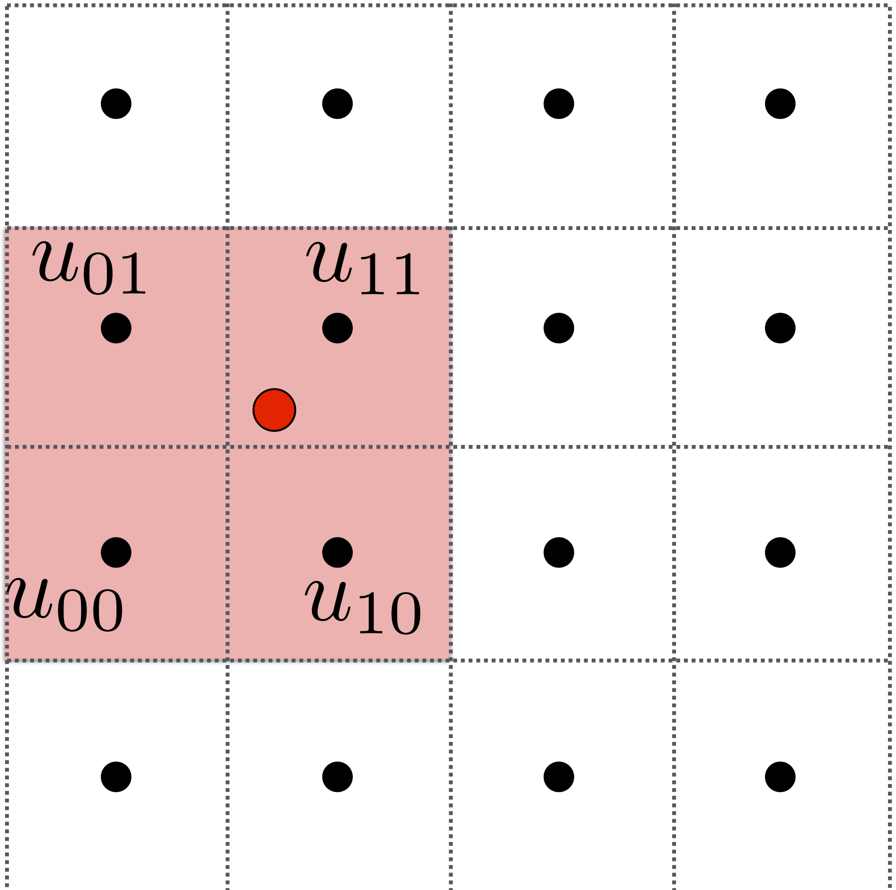
Bilinear Interpolation



Want to sample texture value f(x,y) at red point

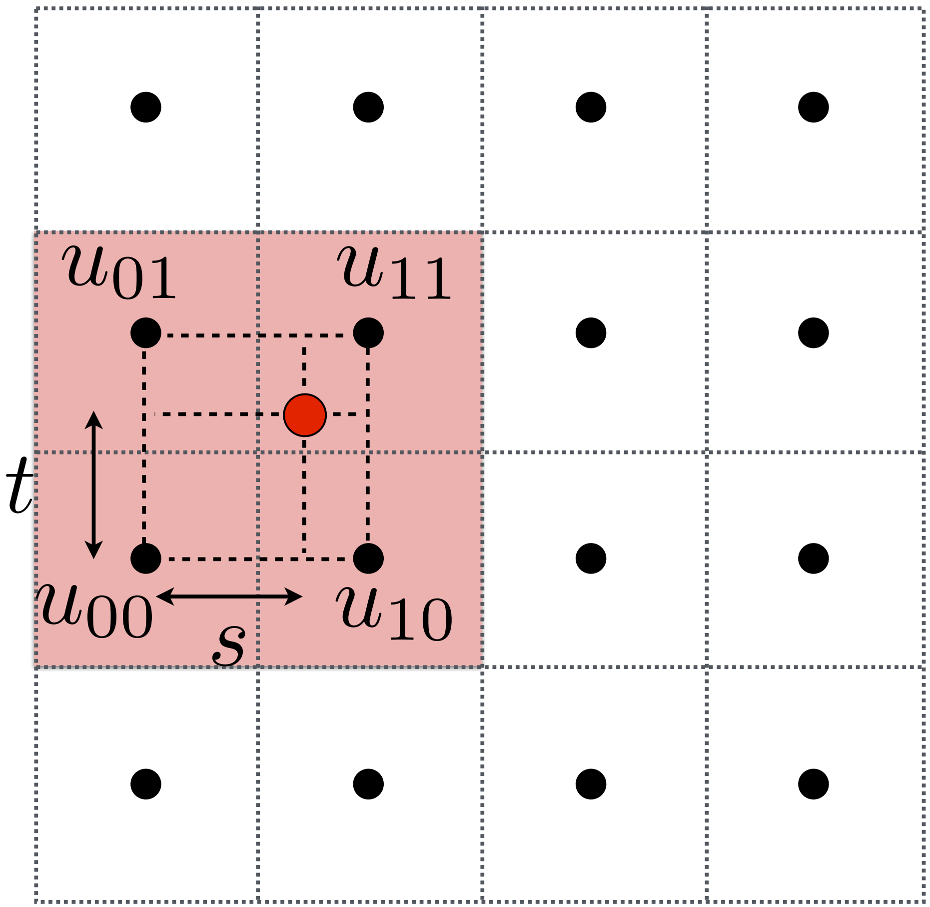
Black points indicate texture sample locations

Bilinear Interpolation



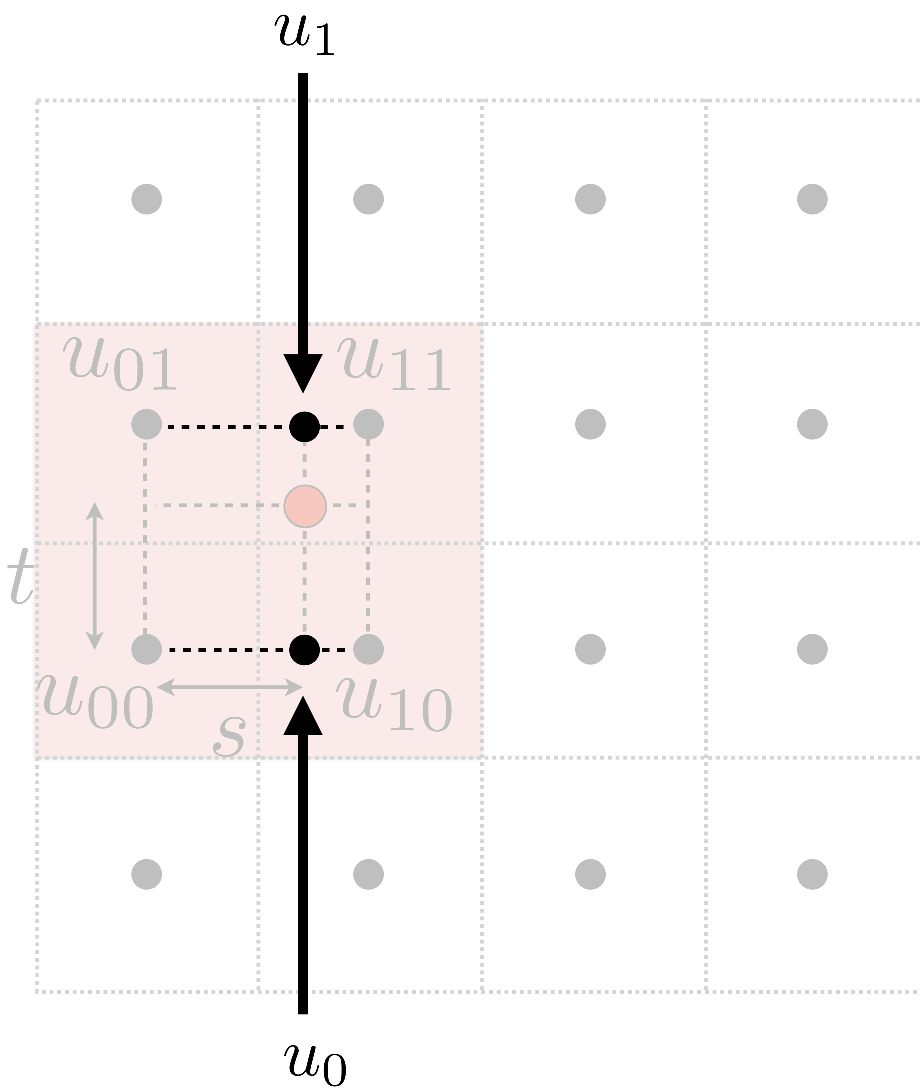
Take 4 nearest sample locations, with texture values as labeled.

Bilinear Interpolation



And fractional offsets, (s,t) as shown

Bilinear Interpolation



Linear interpolation (1D)

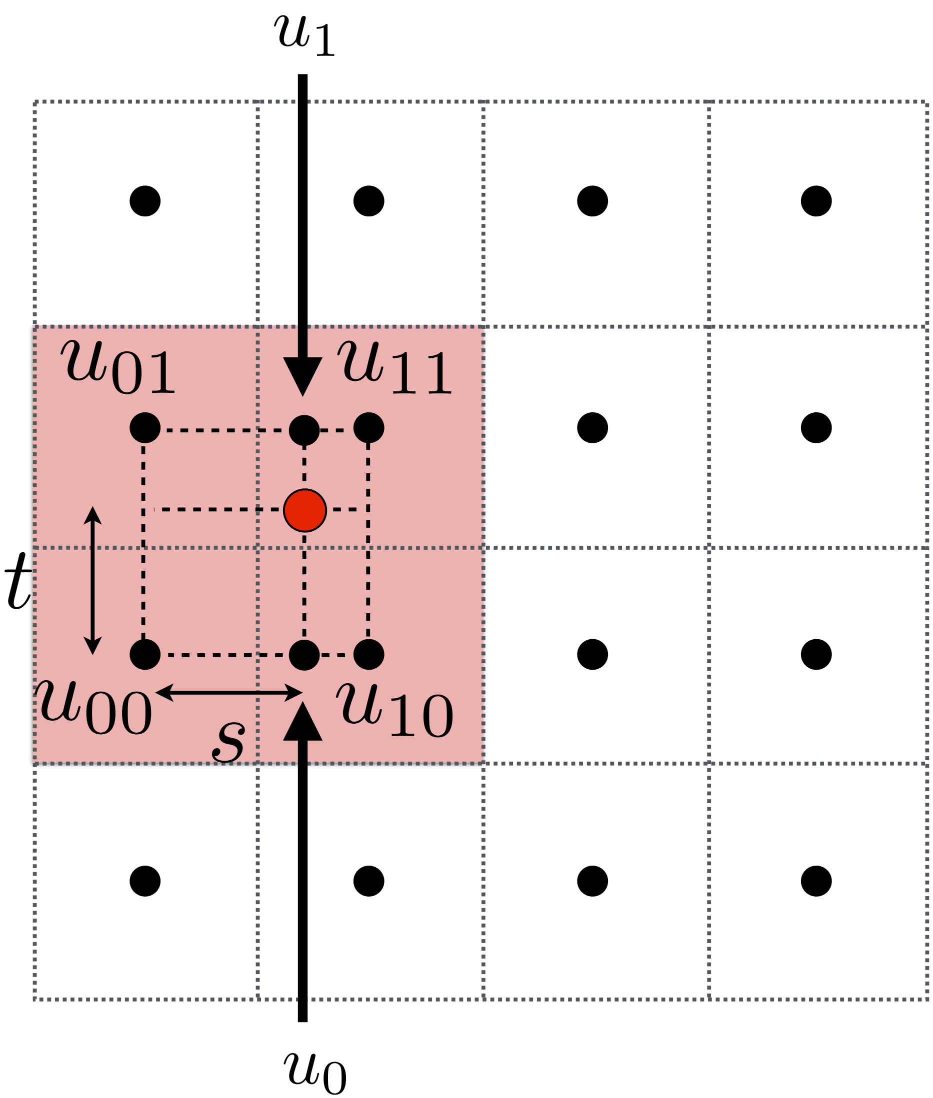
$$\mathrm{lerp}(x, v_0, v_1) = v_0 + x(v_1 - v_0)$$

Two helper lerps (horizontal)

$$u_0 = \mathrm{lerp}(s, u_{00}, u_{10})$$

$$u_1 = \mathrm{lerp}(s, u_{01}, u_{11})$$

Bilinear Interpolation



Linear interpolation (1D)

$$\mathrm{lerp}(x, v_0, v_1) = v_0 + x(v_1 - v_0)$$

Two helper lerps

$$u_0 = \mathrm{lerp}(s, u_{00}, u_{10})$$

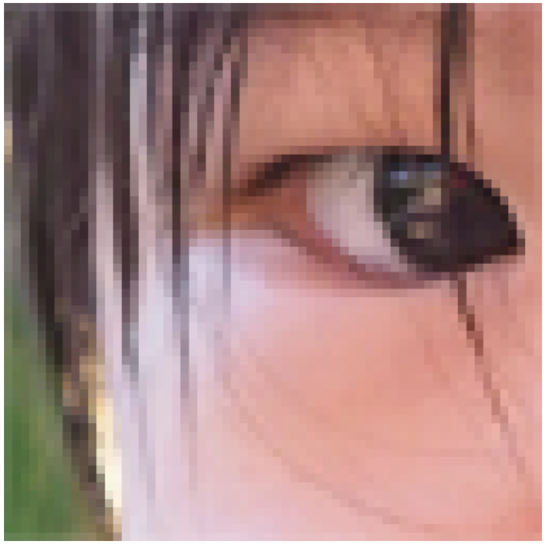
$$u_1 = \mathrm{lerp}(s, u_{01}, u_{11})$$

Final vertical lerp, to get result:

$$f(x, y) = \mathrm{lerp}(t, u_0, u_1)$$

Comparison

Generally bilinear is good enough



Nearest

Bilinear

Bicubic

How to change aspect ratio?

Letterboxing
Scaling

Challenge

Changing aspect ratio causes distortion

Cropping may remove important contents

Content-aware resizing

Seam Carving for Content-Aware Image Resizing

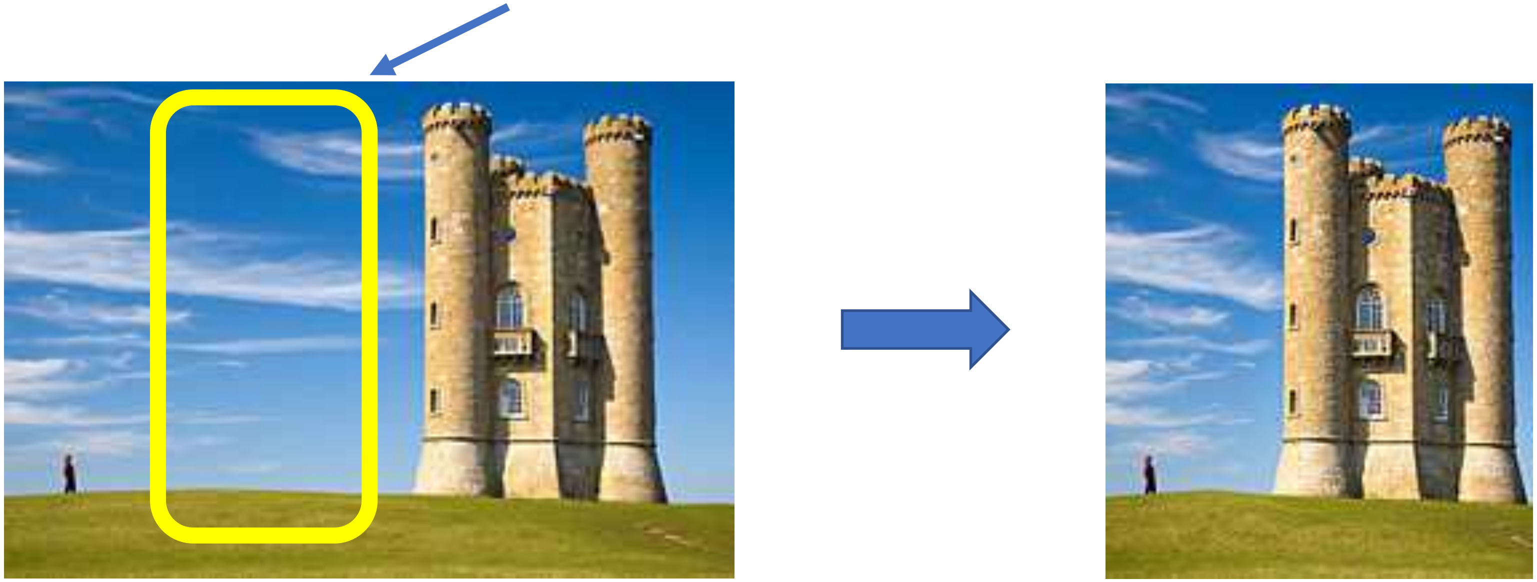
Shai Avidan
Mitsubishi Electric Research Labs

Ariel Shamir
The Interdisciplinary Center & MERL

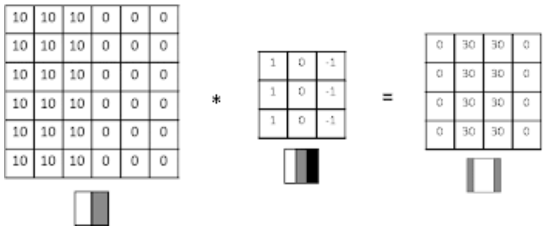
Basic idea

Problem statement: we need to remove n pixels from each row

Basic idea: remove unimportant pixels



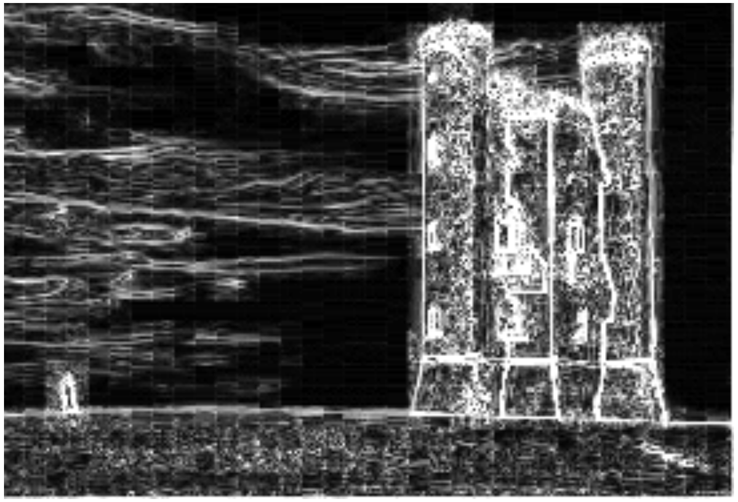
Importance of pixel



How to measure importance of a pixel?

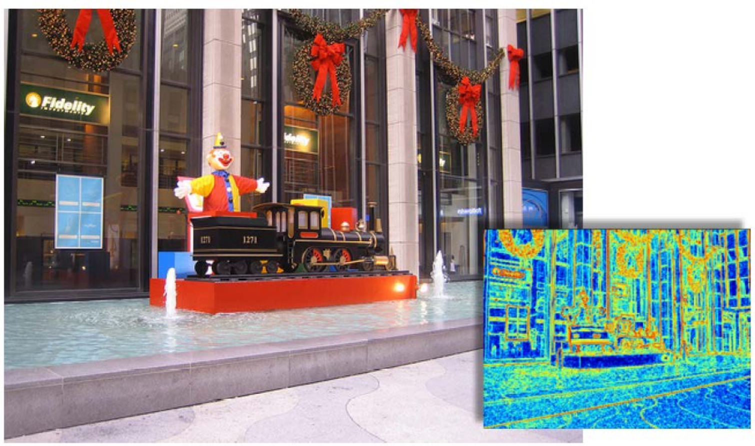
- A simple idea – edges are important
- Edge energy:

$$E(I) = \left|\frac{\partial I}{\partial x}\right| + \left|\frac{\partial I}{\partial y}\right|$$



Greedy algorithm

Remove pixels or columns with the smallest energy?



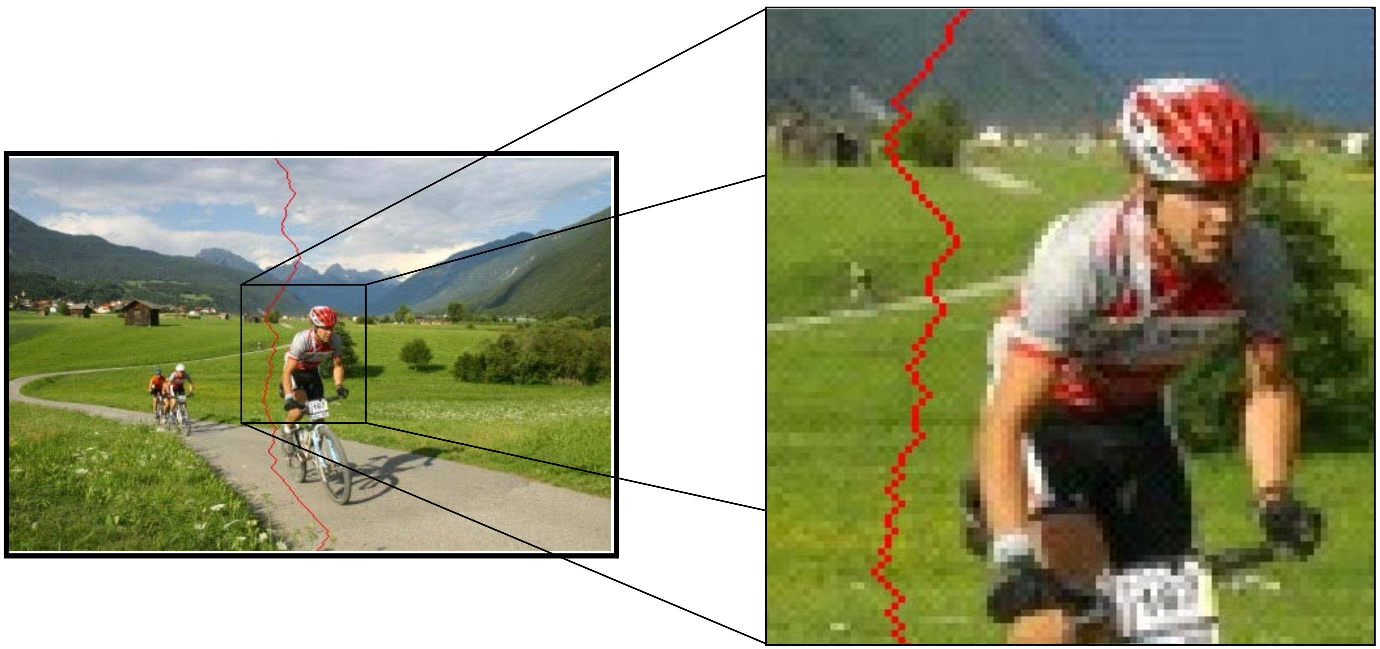
Least-energy pixels

Least-energy columns

Seam carving

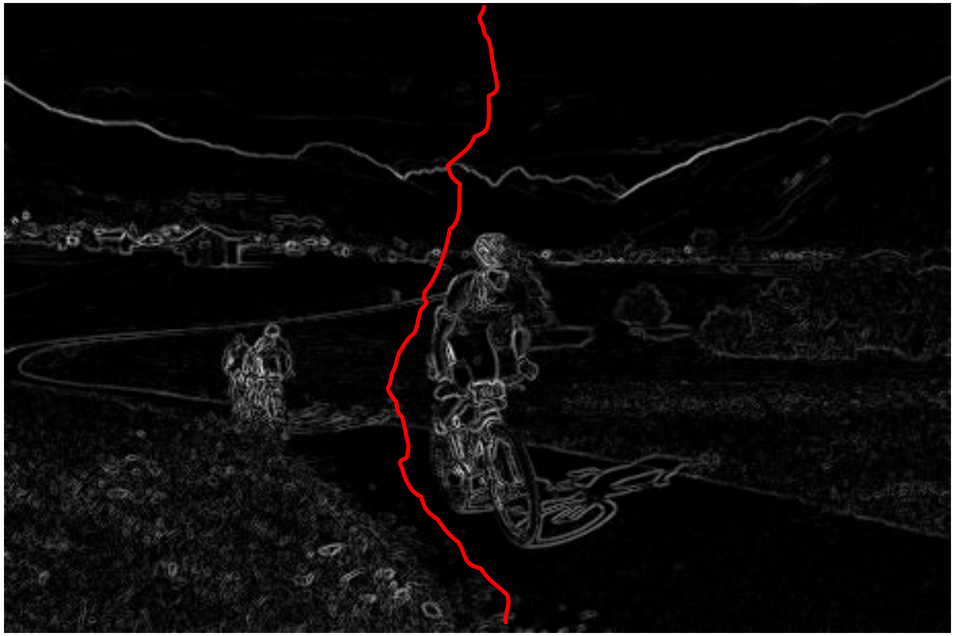
A better solution – seam carving

- Definition of seam: connected path of pixels from top to bottom (or left to right). Exactly one in each row



Finding the seam?

Finding the seam



$$E(\mathbf{I}) = \left|\frac{\partial}{\partial x}\mathbf{I}\right| + \left|\frac{\partial}{\partial y}\mathbf{I}\right| \Rightarrow s^* = \arg\min_s E(s)$$

Finding the seam

Going from top to bottom

- If M(i,j) = minimal cost of a seam going through (i,j)
- Then:

$$\mathbf{M}(i,j) = E(i,j) + \min\left(\mathbf{M}(i-1,j-1), \mathbf{M}(i-1,j), \mathbf{M}(i-1,j+1)\right)$$

- Solved by dynamic programing

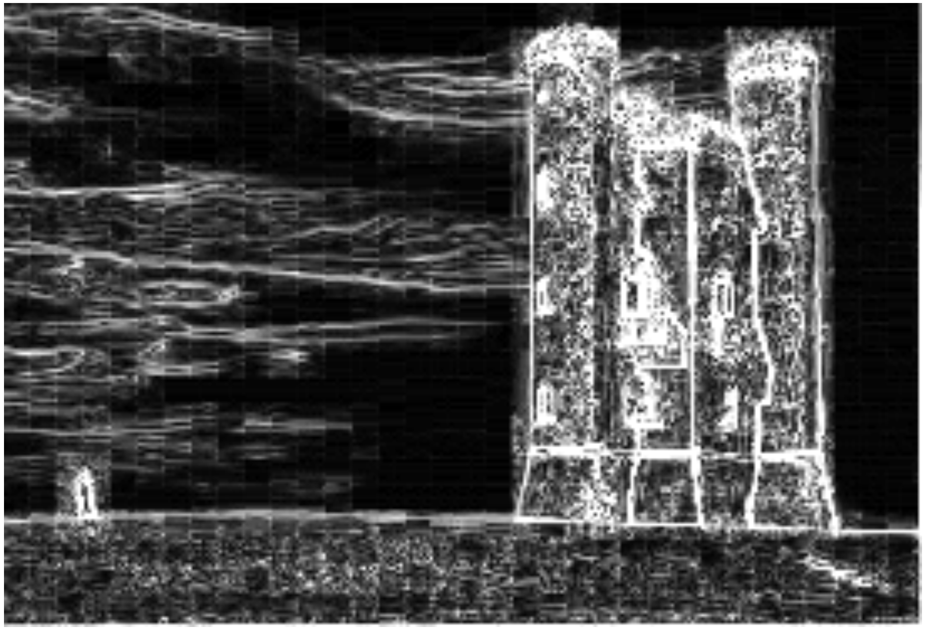
5	8	12	3
9	2	3	9
7	3	4	2
4	5	7	8

Seam carving algorithm

- Starting with an image such as

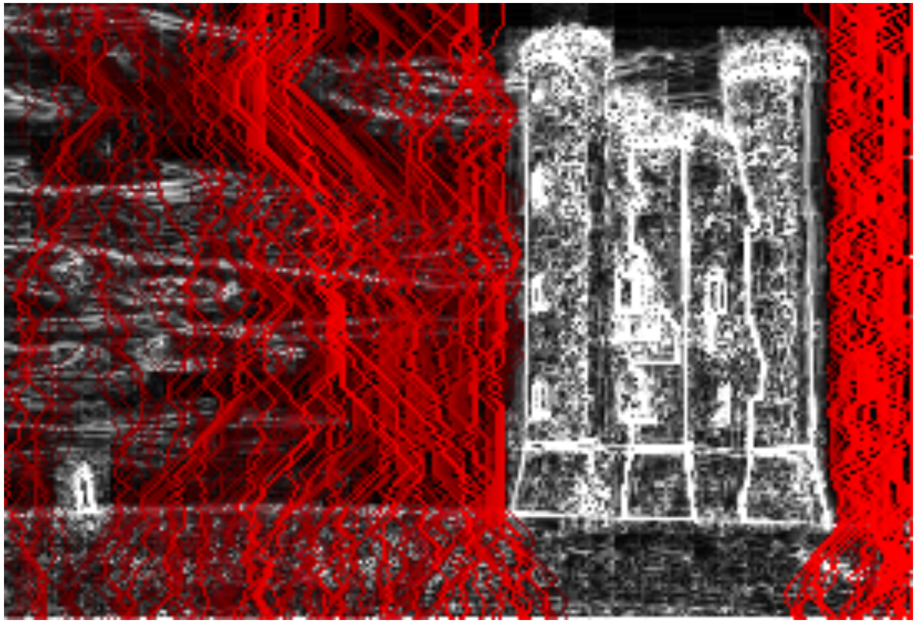
Seam carving algorithm

- The weight/density/energy of each pixel is then calculated



Seam carving algorithm

- Seams can then be calculated and ranked via the dynamic programming



Seam carving algorithm

- Then the seams are removed from the image

Results

Original

Seam Carving

Scaling

Results

Cropping

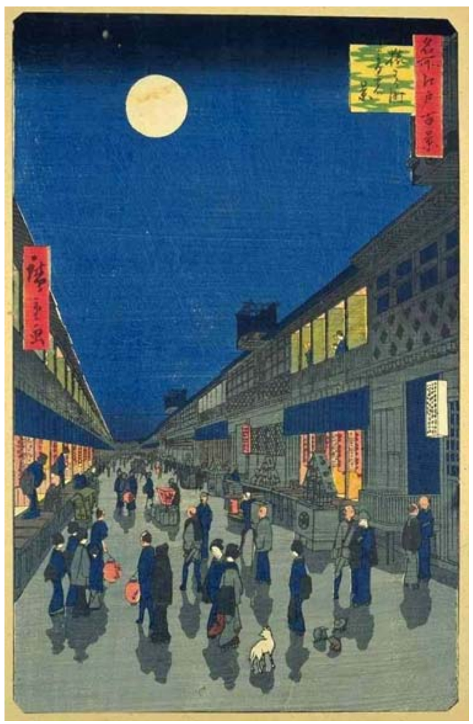
Seams

Scaling

Seam insertion

Can we enlarge an image?

- Basic idea: reverse the seam carving process




Seam insertion

Find k seams to insert

Then interpolate pixels


Shai Avidan
Mitsubishi Electric Research Lab
Ariel Shamir
The interdisciplinary Center & MERL

Super-Resolution

Original

Bi-Cubic

Super-Resolution

Super-Resolution

Goal

- Produce a detailed, realistic output image.
- Be faithful to the low resolution input image.

Basic idea

- Build some statistical model of image.
- Enforce an up-sampled image to obey those statistics.

Types of methods

- Exemplar based – a collection of examples as the "image model"
- Optimization based – mathematical image model
- Deep learning – learned using deep neural networks

Example-Based Super-Resolution

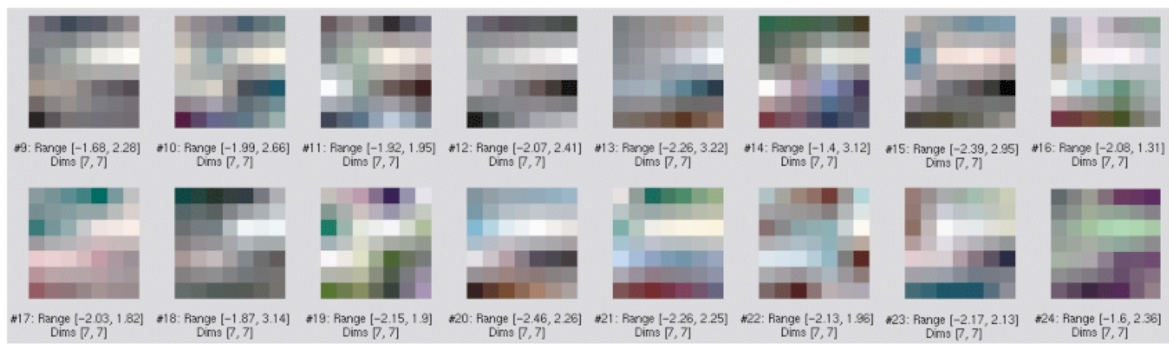
William T. Freeman, Thouis R. Jones, and Egon C. Pasztor
Mitsubishi Electric Research Labs

Basic idea

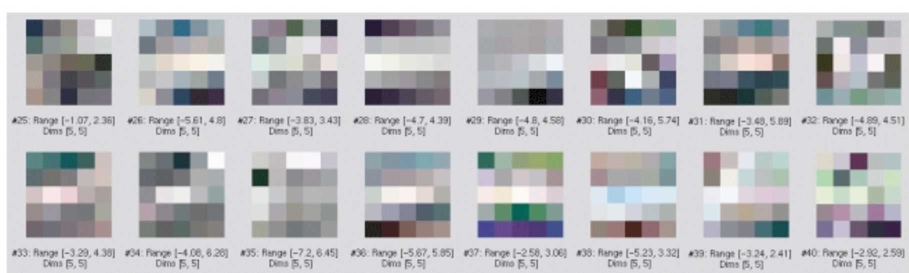
Replace low-res image patches in the input image with high-res patches from a database

Input patch

Closest image patches from database



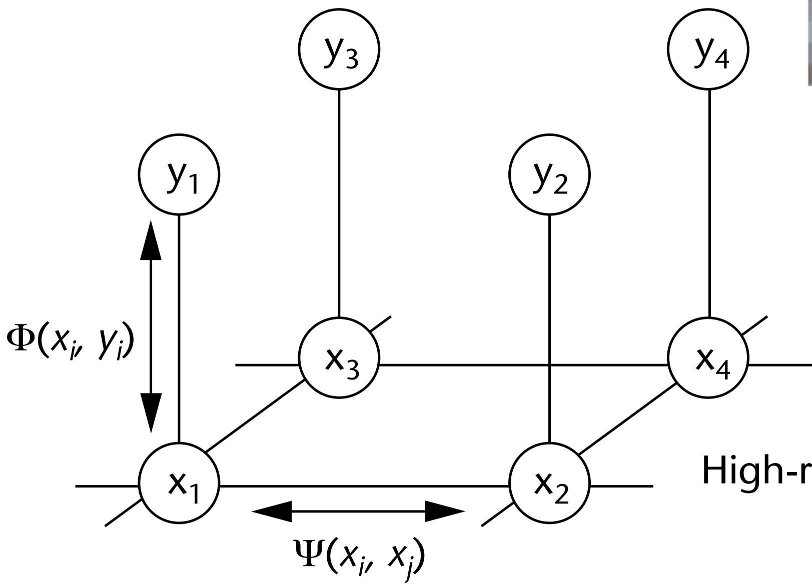
Corresponding high-resolution patches from database

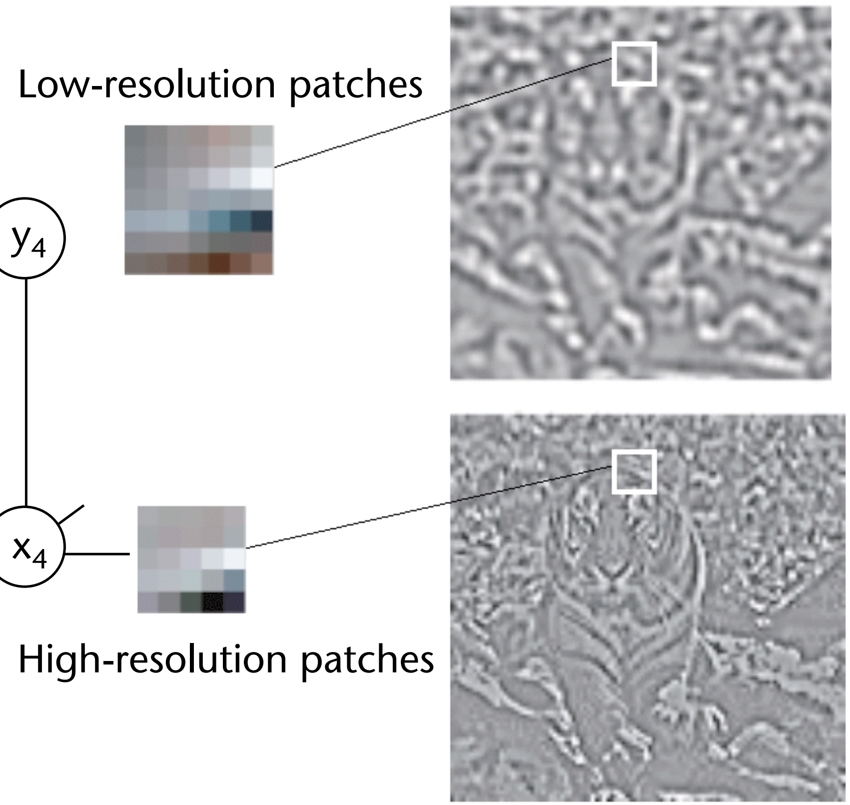


MRF optimization

Modeling smoothness

$$P(x \mid y) = \frac{1}{Z} \prod_{(ij)} \psi_{ij}(x_i, x_j) \prod_i \phi_i(x_i, y_i)$$





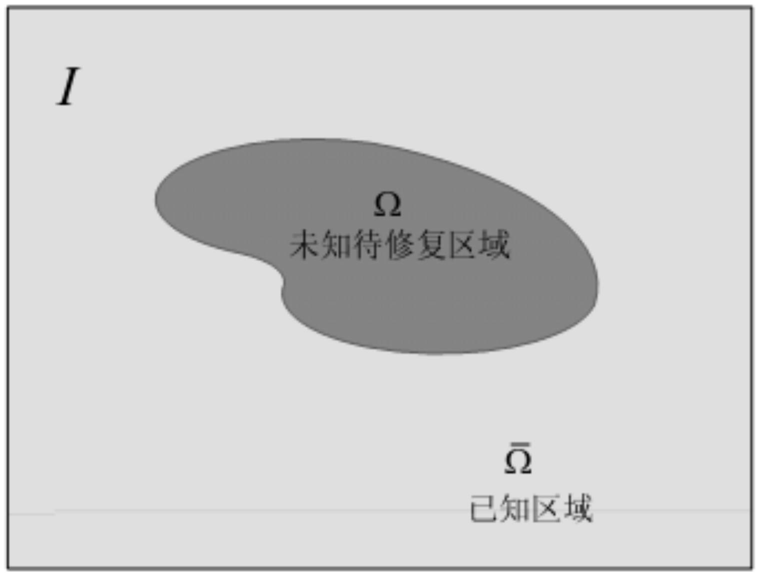
Results

Image completion

Restoration

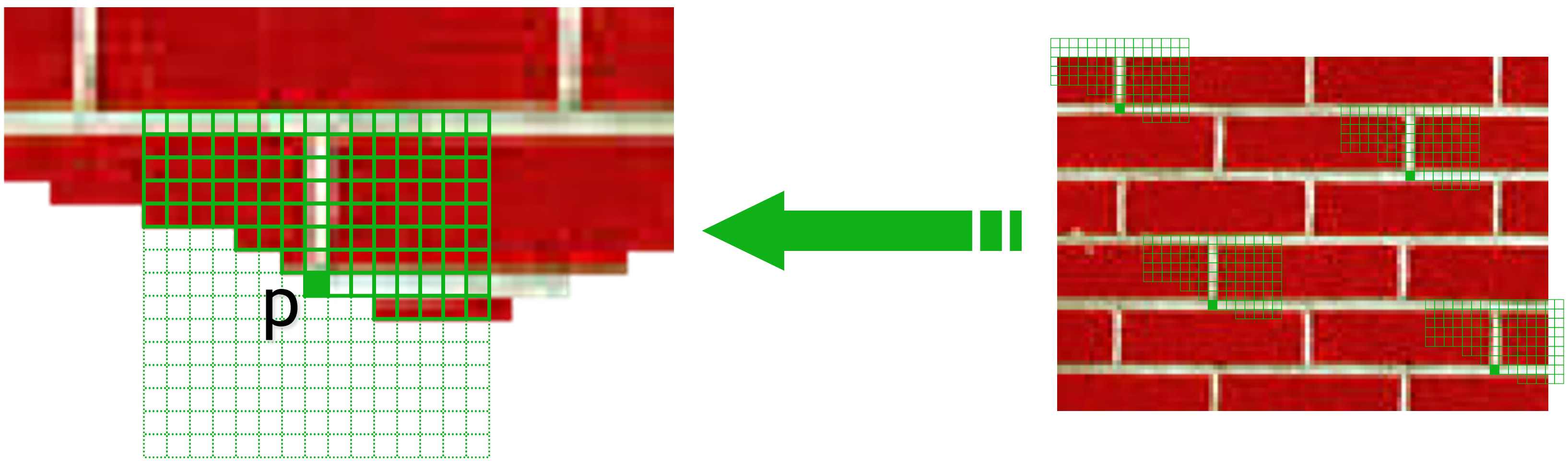
Object
removal

Problem statement



I 代表待修复图像，I 中深色区域 Ω 代表受损区域，也就是需要修补的区域，其余部分 $\overline{\Omega} = I - \Omega$ 为已知区域。Completion 即根据已知区域 $\overline{\Omega}$ 修复未知区域，得到重建区域 Ω'，使得修复后的图像 $I' = \Omega' \cup \Omega$ 在视觉上自然

Examplar-based methods

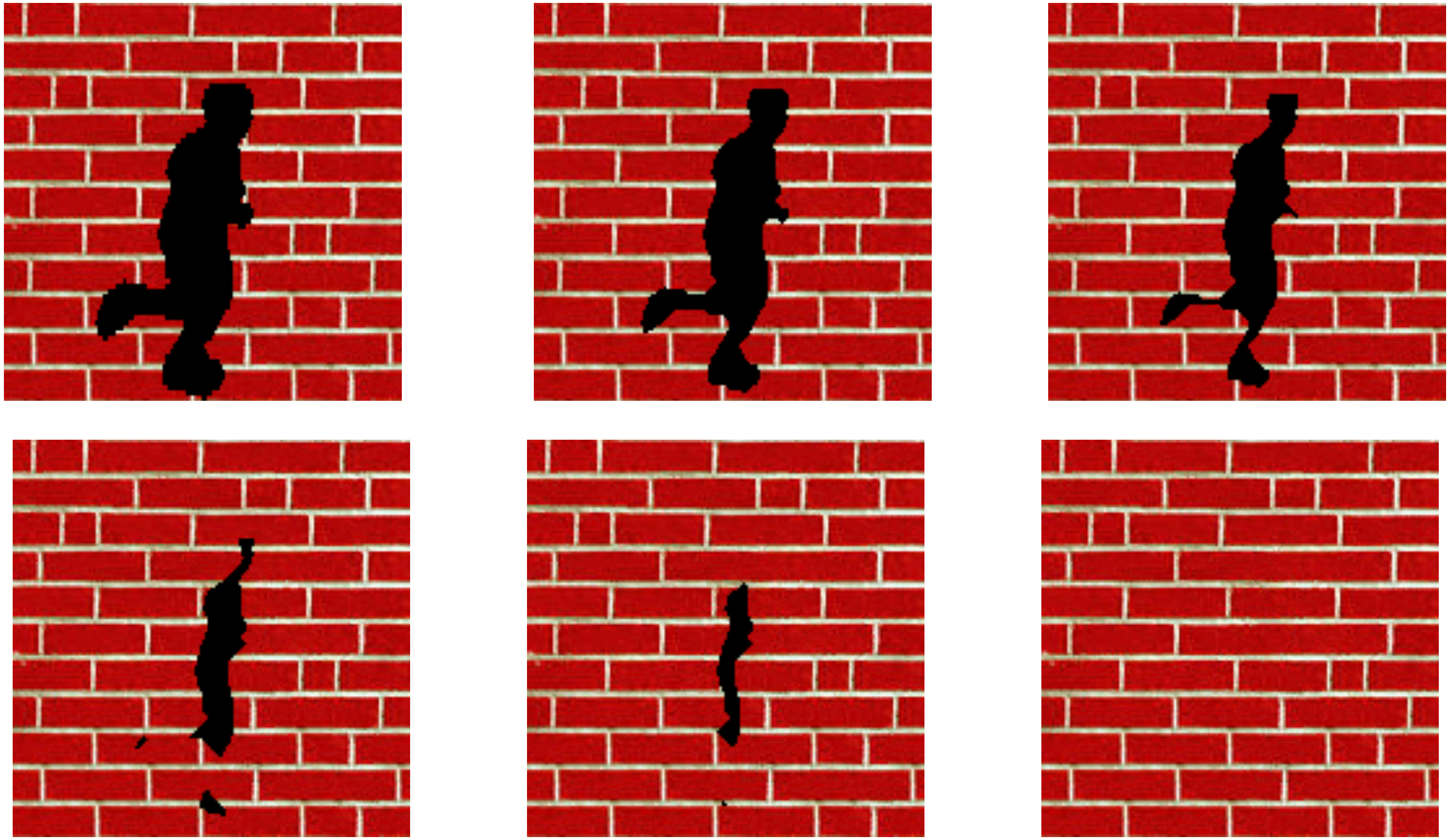


空洞的边界

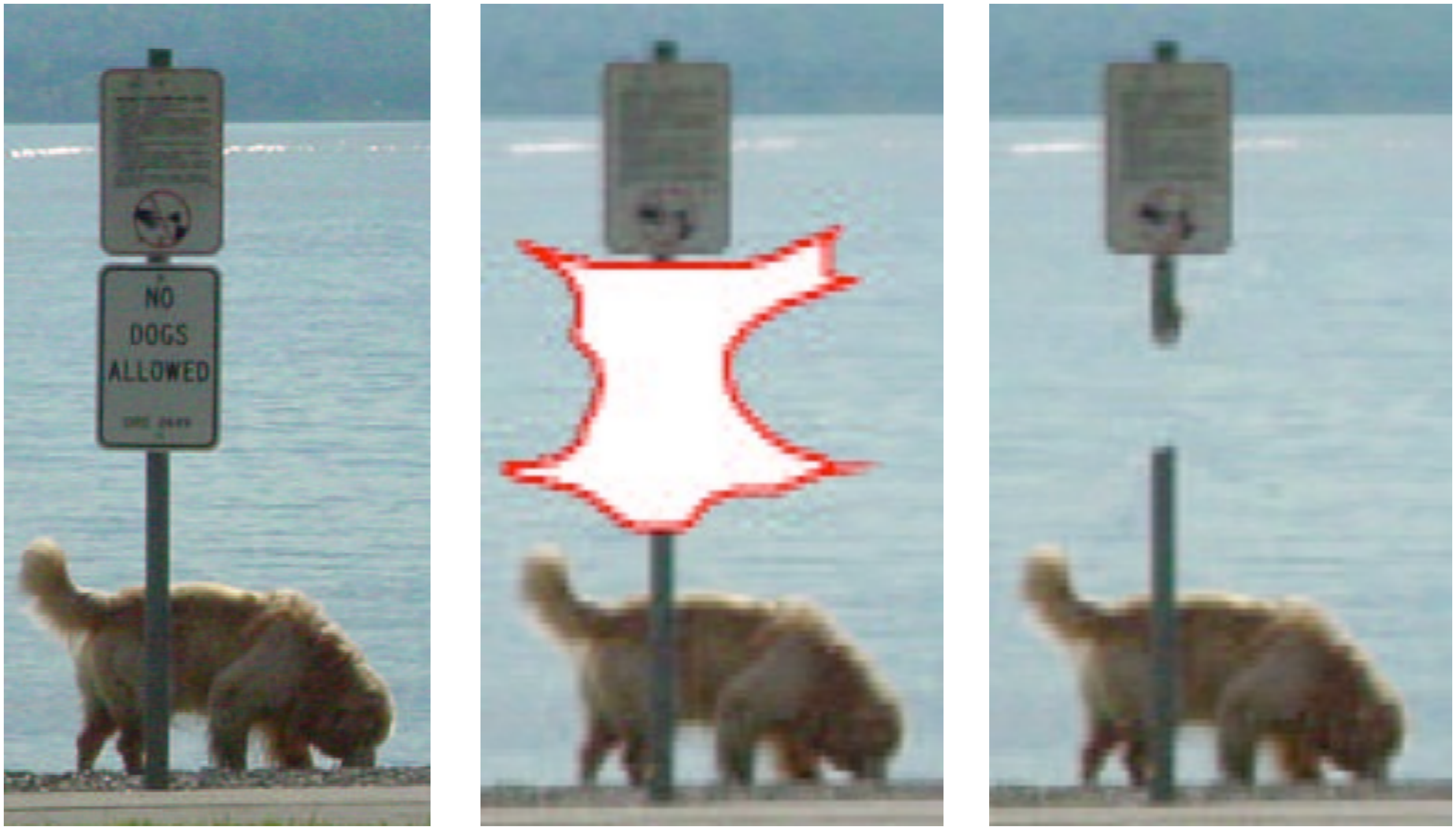
已知的样本区域

Examplar-based methods

"剥洋葱"



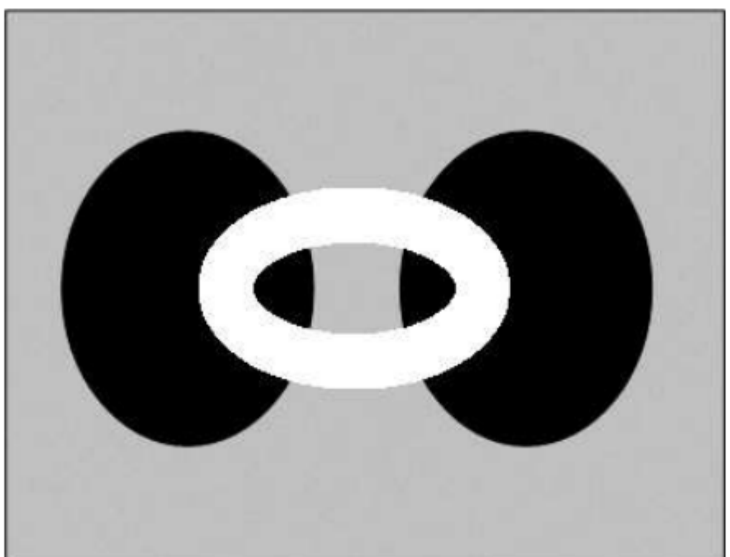
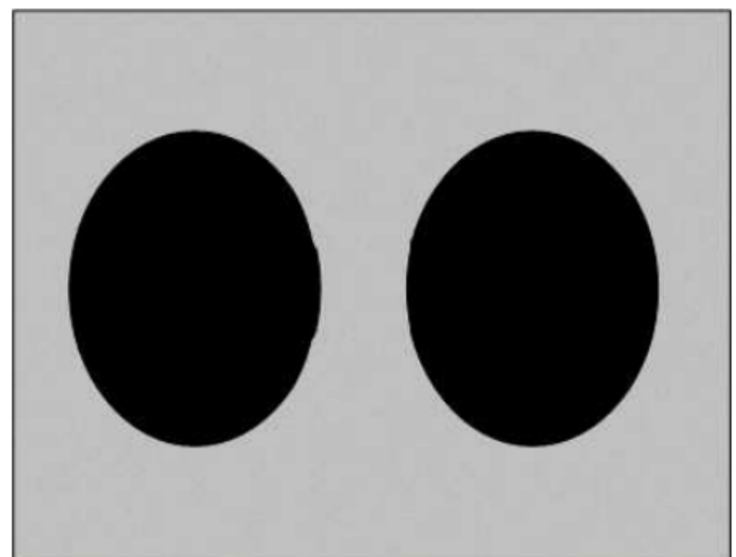
The order matters!



带有优先级的填充策略

Image Completion by Example-Based Inpainting

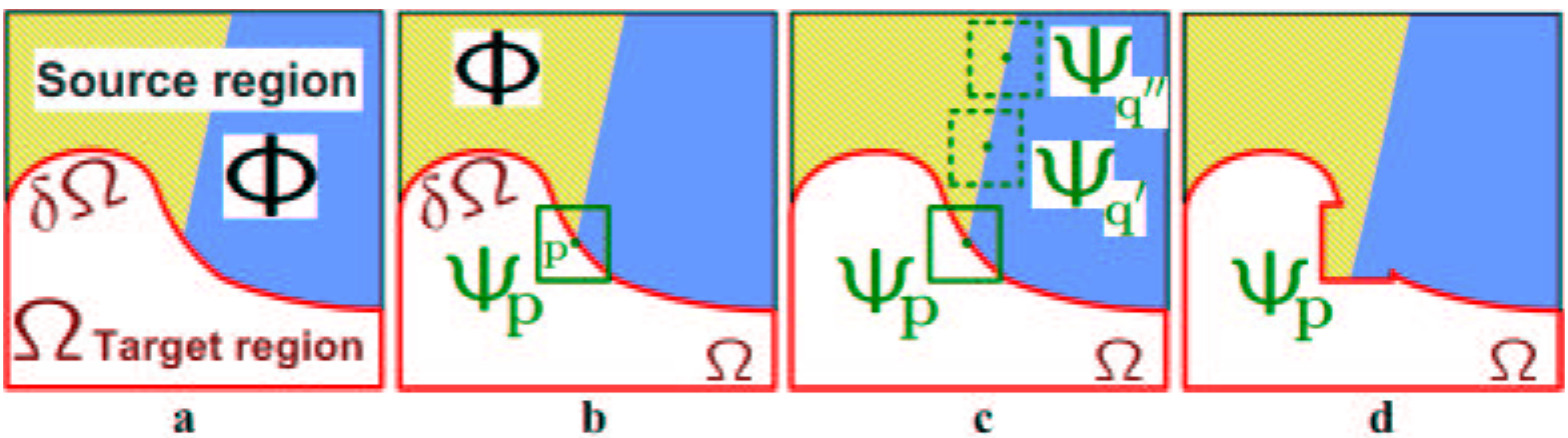
A. Criminisi, P. Perez, and K. Toyama,
CVPR 2003

带有优先级的填充策略

算法概览

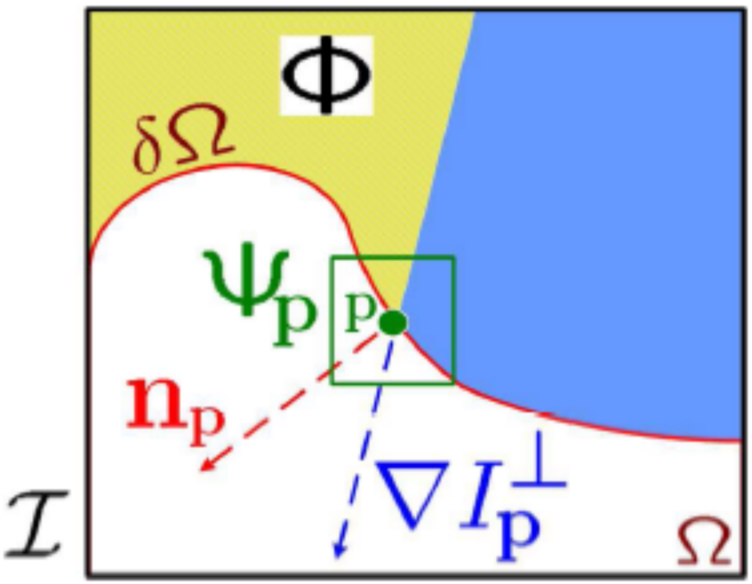
1. 用户选择需要补全的区域
2. 确定目前空洞边缘的像素位置
3. 为每一个像素计算优先级权重
4. 查找到优先级权重最大的像素位置p, 并确定对应的块P
5. 从图像已知区域匹配出最相似的块S, 对P中不可见的像素进行补全
6. 更新优先级权重
7. 重复2-6步骤, 直到所有的像素被修复



带有优先级的填充策略

- 优先级度量:

$$P(\mathbf{p}) = C(\mathbf{p})D(\mathbf{p})$$



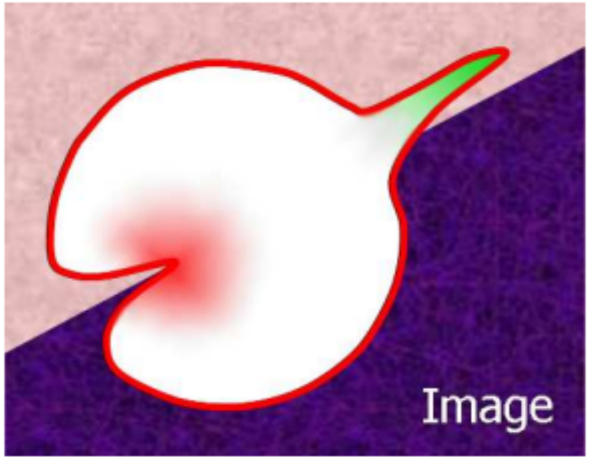
置信度项: 待填充块中有多少已知像素

$$C(\mathbf{p}) = \frac{\sum_{\mathbf{q}\in\Psi_{\mathbf{p}}\cap\bar{\Omega}} C(\mathbf{q})}{|\Psi_{\mathbf{p}}|}$$

数据项: 希望顺着边缘填充（保结构）

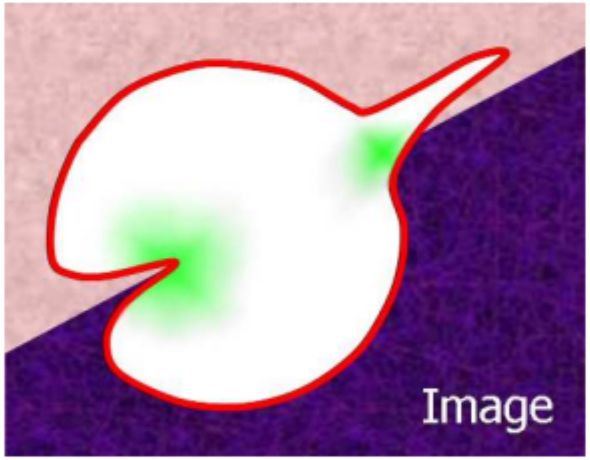
$$D(\mathbf{p}) = \frac{|\nabla I_{\mathbf{p}}^{\perp}\cdot\mathbf{n}_{\mathbf{p}}|}{\alpha}$$

带有优先级的填充策略



a

a图显示置信度的分布, 绿色表示置信度高的区域
红色表示相对较低的取悦



b

b图显示数据项的分布, 绿色表示置信度高的区域

实验结果

a

b

c

d

e

f

实验结果

a

b

实验结果

a

b

c

d

实验结果

a

b

基本问题: 结构vs纹理

结构信息的补全比纹理信息的补全要困难的多, 能否通过添加一些交互来解决这个问题?

带交互的补全算法

Image Completion with Structure Propagation

J. Sun, L. Yuan, J. Jia, and H. Shum

SIGGRAPH 2005

带有优先级的填充策略

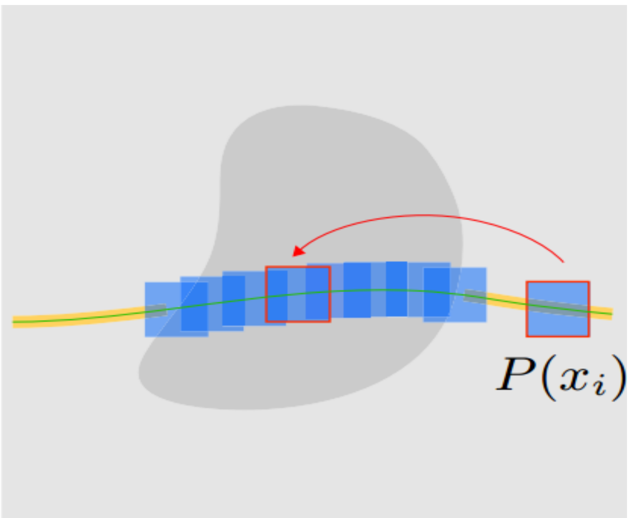
算法概览

1. 用户输入：用户在空洞区域以及已知图像区域勾画结构线,
2. 结构补全：该算法在已知图像区域采样, 通过优化一个目标能量来决定如何将样本填充被结构线覆盖的空洞区域
3. 纹理补全：补全剩余区域的纹理
4. 光测度修正

目标能量

对于每一个锚点 p_i 我们找到一个标签 $x_i \in \{1,2,\ldots,N\}$ 对应于其中的一个样本块，

将样本块 $P(x_i)$ 复制到 p_i 的位置如下图所示。



能量函数定义如下:

$$E(X)=\sum_{i\in v}E_1(x_i)+\sum_{(i,j)\in\varepsilon}E_2(x_i,x_j),$$

$$E_1(x_i)=k_sE_s(x_i)+k_iE_I(x_i).$$

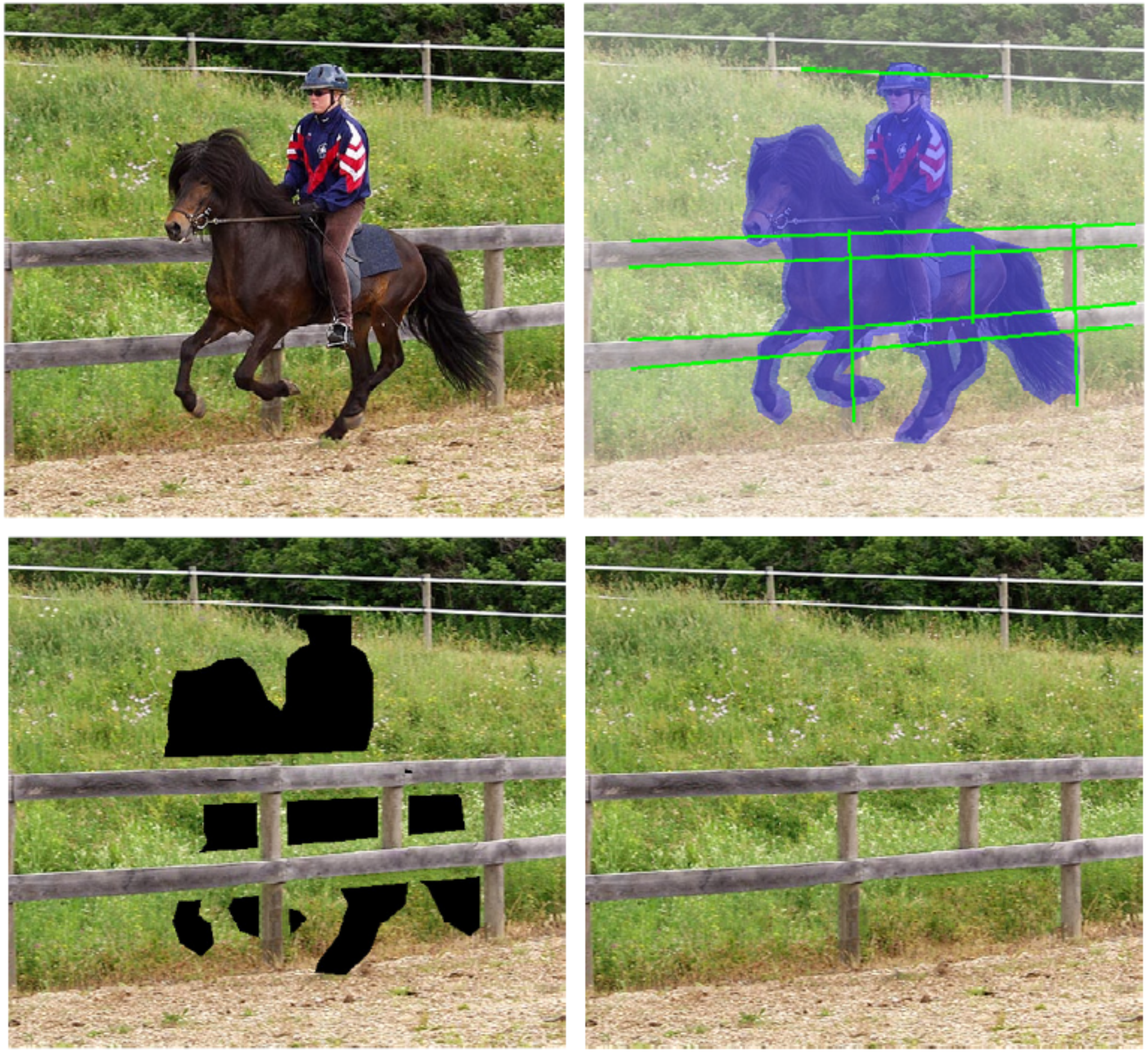
$E_s(x_i)$，$E_I(x_i)$ 和 $E_2(x_i,x_j)$ 分别表示结构，边界和一致性约束。

实验结果

实验结果

Criminisi等人的方法

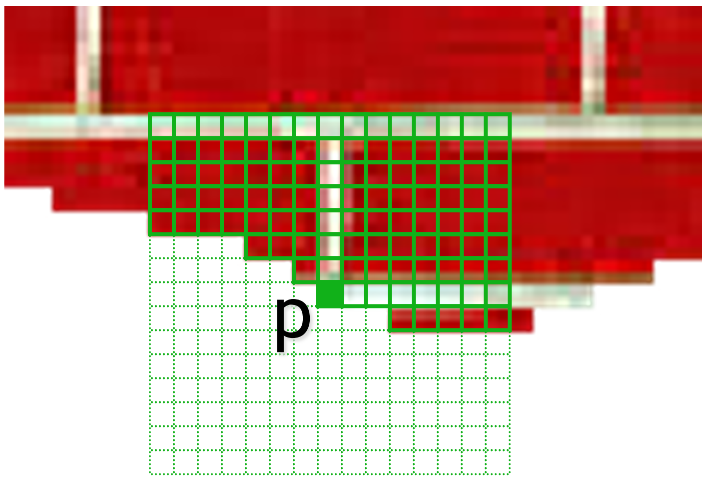
实验结果



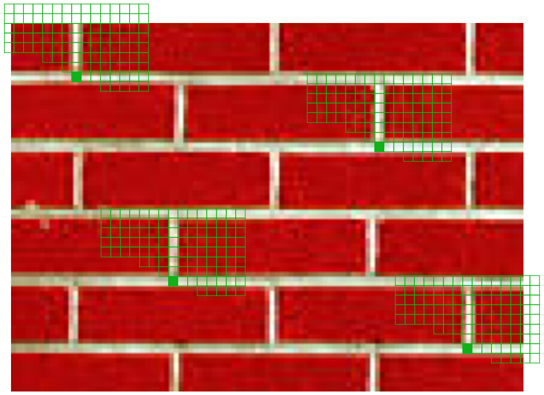
实验结果

Criminisi等人的方法

Recap: examplar-based methods



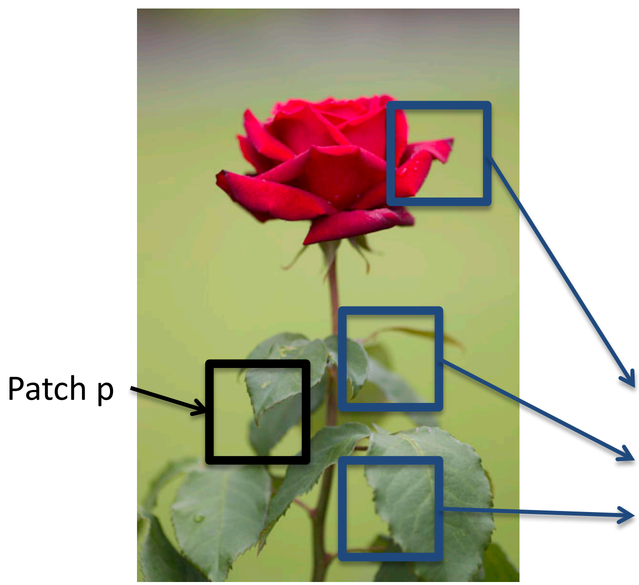
空洞的边界



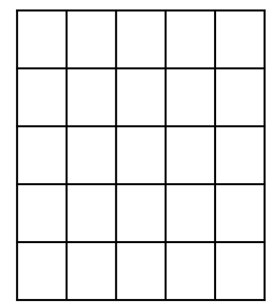
已知的样本区域

How to solve patch match?

Many algorithms need to search the most similar patches



Which patch is most similar?



Patch p with 25 dimensions

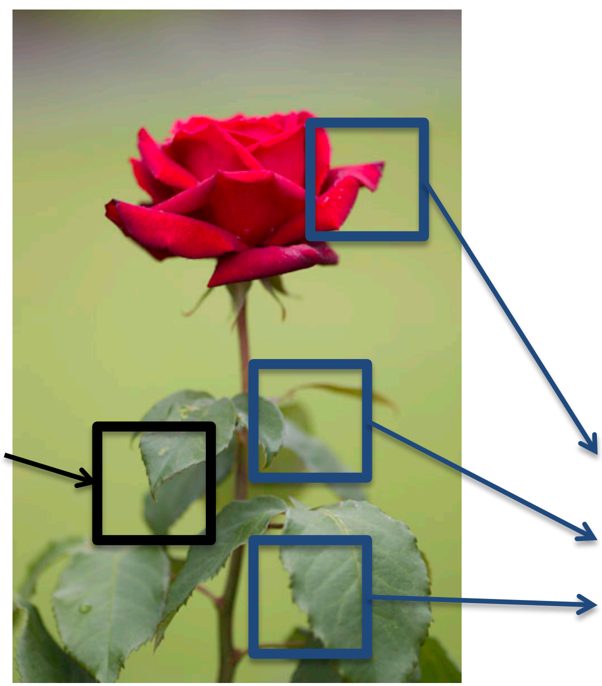
Patch match

A naïve searching algorithm

Sample every possible patch
to find best match!

$O(mM^2)$

Patch p



Which patch is most similar?

快速的图像块匹配算法

PatchMatch: A randomized correspondence algorithm for structural image editing

Barnes, C., Shechtman, E., Finkelstein, A
SIGGRAPH 2009

Slides credit: Jiamin Bai
http://vis.berkeley.edu/courses/cs294-69-fa11/wiki/images/1/18/05-PatchMatch.pdf

PatchMatch algorithm

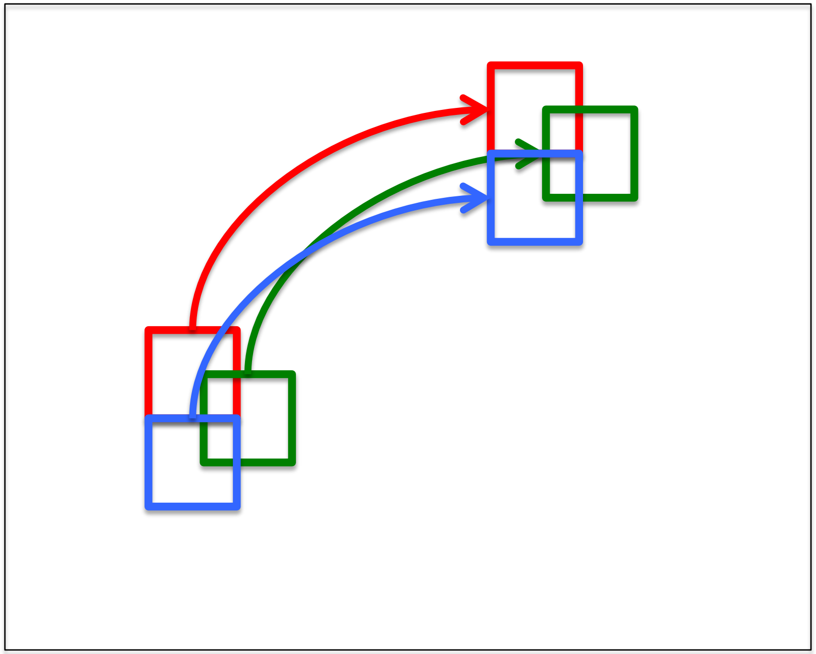
Key ideas:

- Neighboring pixels have coherent matches
- Large number of random sampling will yield some good guesses.

Key idea

- Offset: dispalcement from source to targe
- Neighbors have similar offsets

Coherent matches with neighbors



Key idea

Large number of random sampling will yield some good guesses

$\boldsymbol{M}$ number of total pixels

Probability of correct random guess: $1/\boldsymbol{M}$

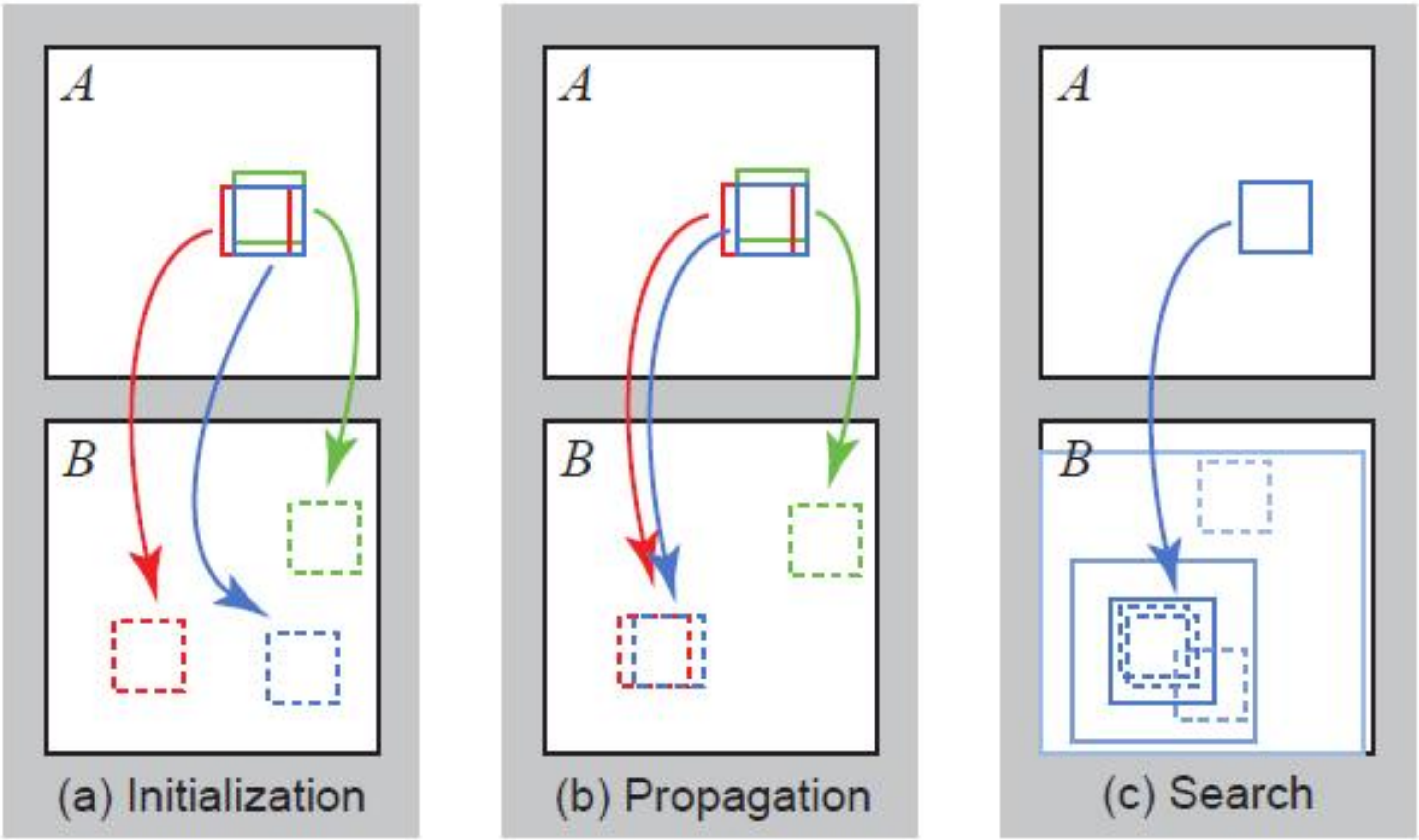
Probability of incorrect random guess: $1 - 1/\boldsymbol{M}$

Probability of all pixels with incorrect guess: $(1 - 1/\boldsymbol{M})^{\boldsymbol{M}}$ [approximately 0.37]

$\Rightarrow$Probability of at least 1 pixel with correct guess : $1 - (1 - 1/\boldsymbol{M})^{\boldsymbol{M}}$

$\Rightarrow$Probability of at least 1 pixel with good enough guess: $1 - (1 - C/\boldsymbol{M})^{\boldsymbol{M}}$

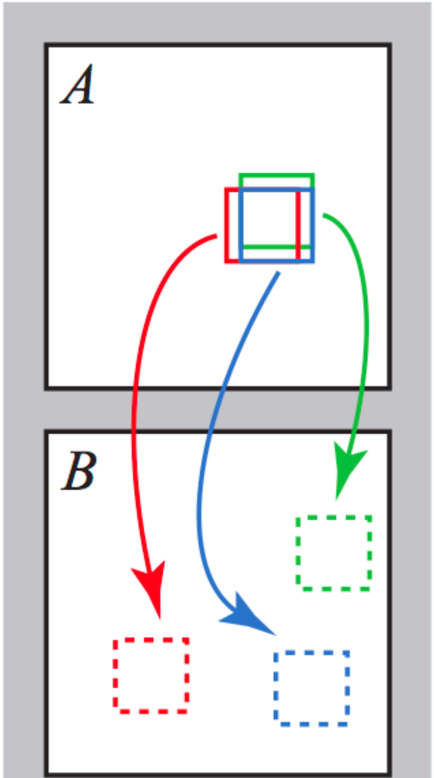
Algorithm: 3 steps



(a) Initialization (b) Propagation (c) Search

Step 1

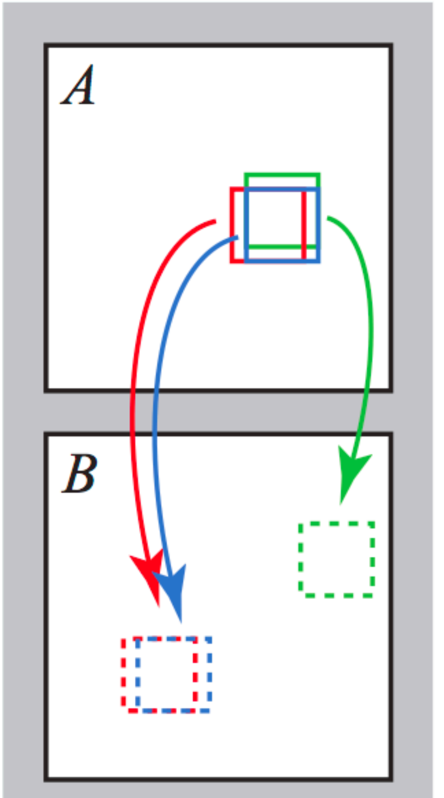
- Each pixel is given a random patch offset as initialization



(a) Initialization

Step 2

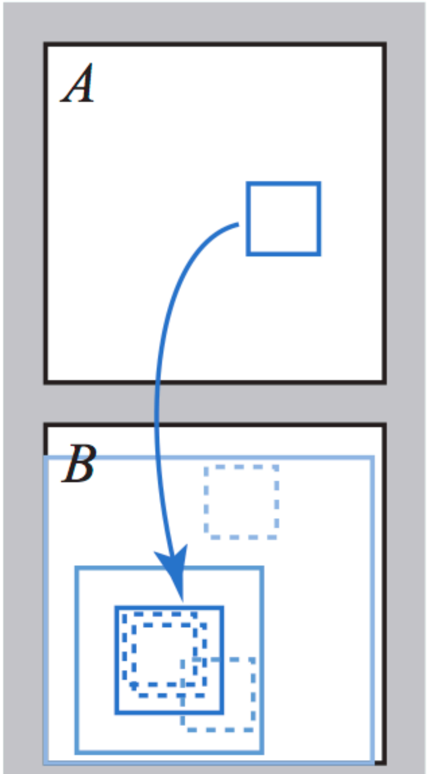
- Each pixels checks if the offsets from neighboring patches give a better matching patch. If so, adopt neighbor's patch offset.



(b) Propagation

Step 3

- Each pixels searches for better patch offsets within a concentric radius around the current offset.
- The search radius starts with the size of the image and is halved each time until it is 1.



(c) Search

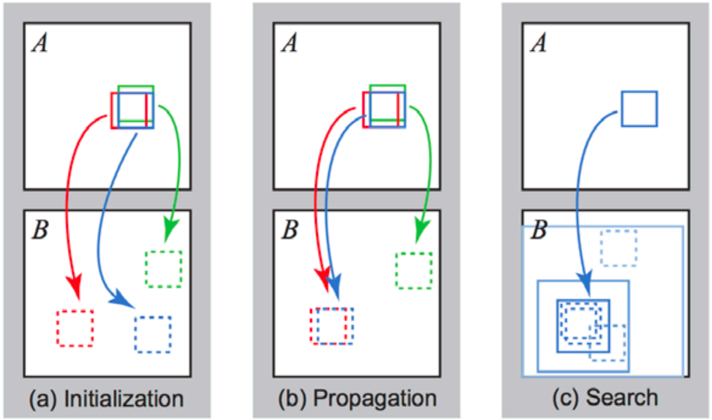
Full algorithm

1. Initialize pixels with random patch offsets

2. Check if neighbors have better patch offsets

3. Search in concentric radius around the current offset for better better patch offsets

4. Go to Step 2 until converge.



O(mMlogM)

Results

(a) originals (b) random (c) $\frac{1}{4}$ iteration (d) $\frac{3}{4}$ iteration (e) 1 iteration (f) 2 iterations (g) 5 iterations

	Time [s]		Memory [MB]	
Megapixels	Ours	*k*d-tree	Ours	*k*d-tree
0.1	0.68	15.2	1.7	33.9
0.2	1.54	37.2	3.4	68.9
0.35	2.65	87.7	5.6	118.3

Using local pathces may be insufficient

Criminisi et al. result

利用更多大数据

Scene Completion Using Millions of Photographs

James Hays and Alexei A. Efros

SIGGRAPH 2007

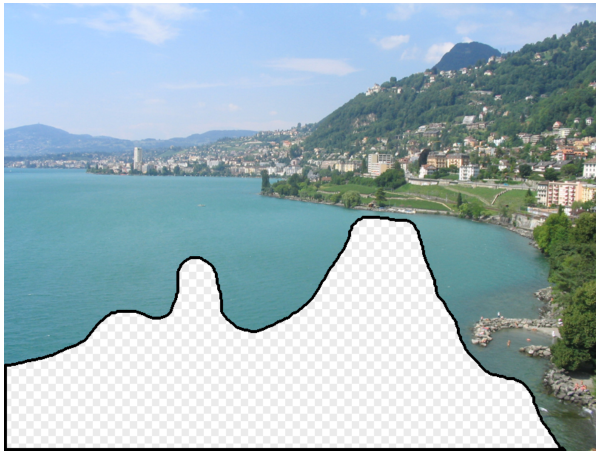
Scene Matching for Image Completion

Data

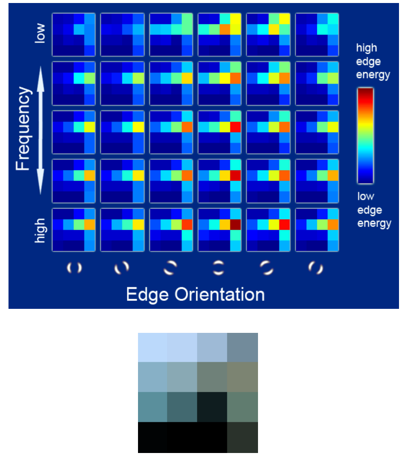
2.3 Million unique images from Flickr groups and keyword searches.

Scene Completion Result

The Algorithm



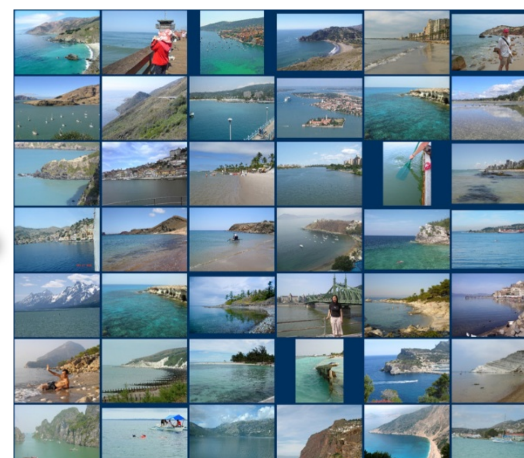
Input image



Scene Descriptor

...

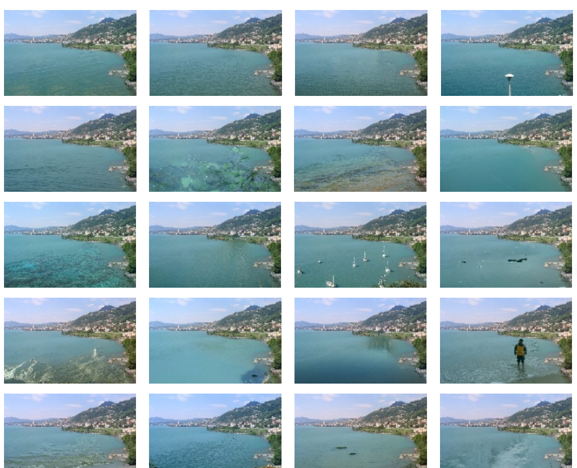
Image Collection



...

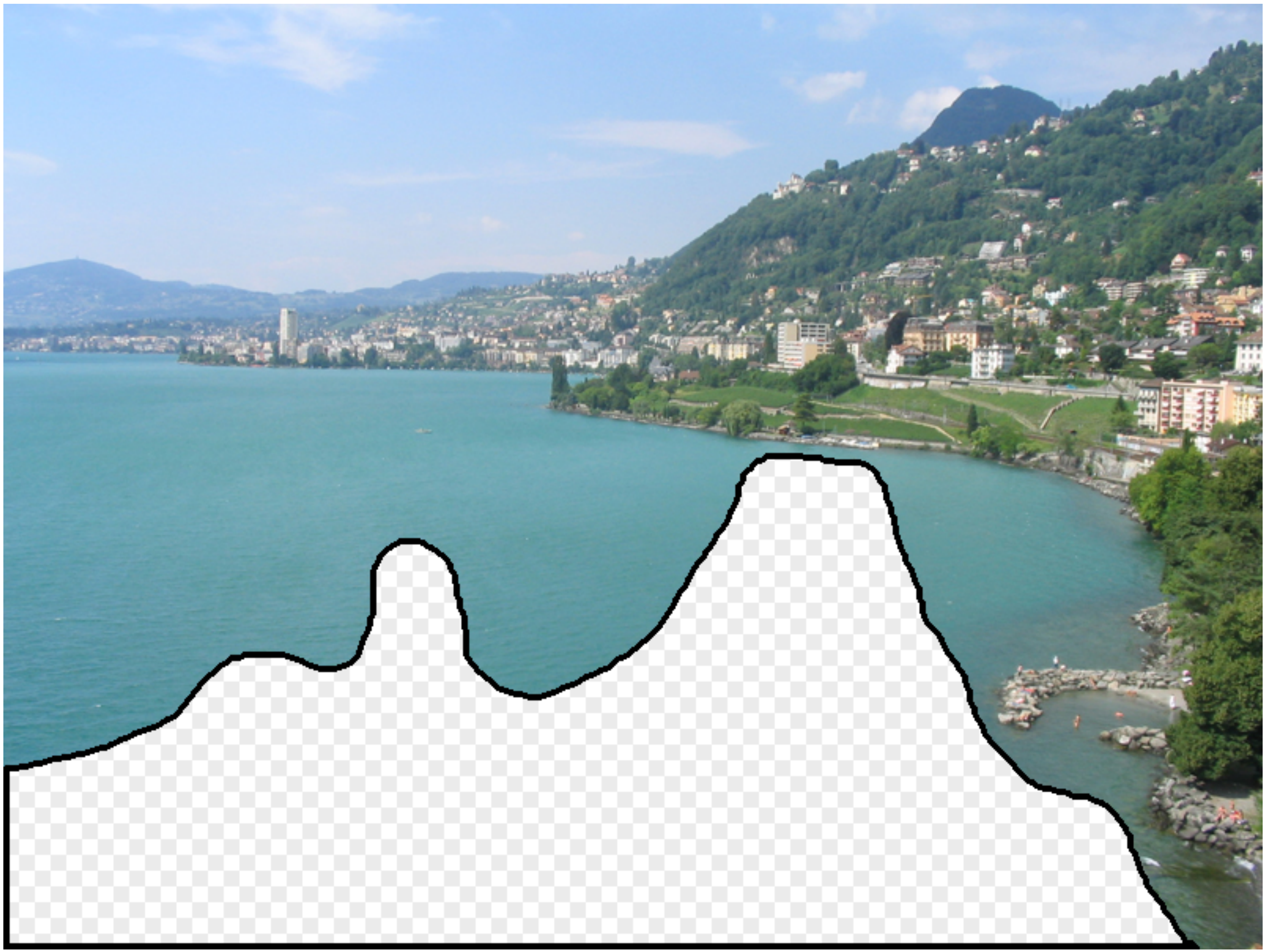
200 matches

Context matching
+ blending

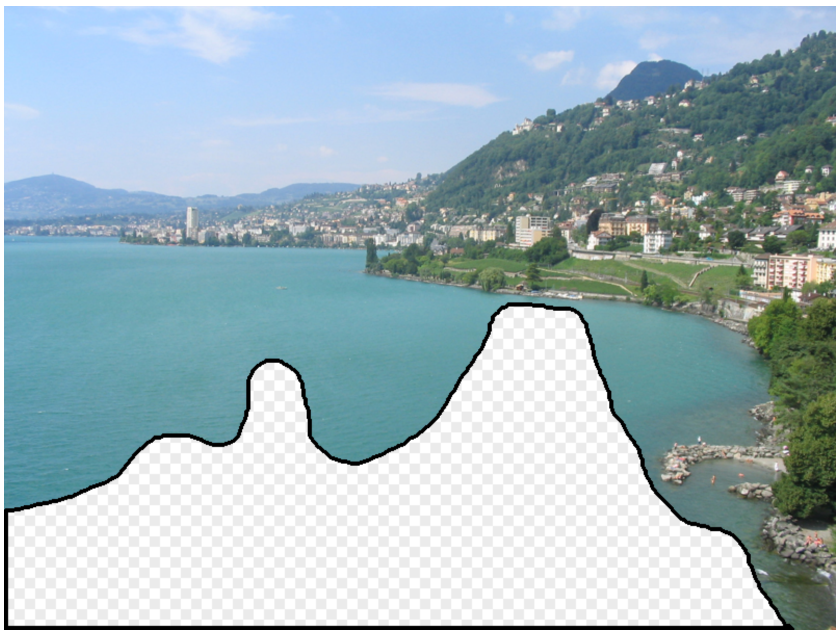


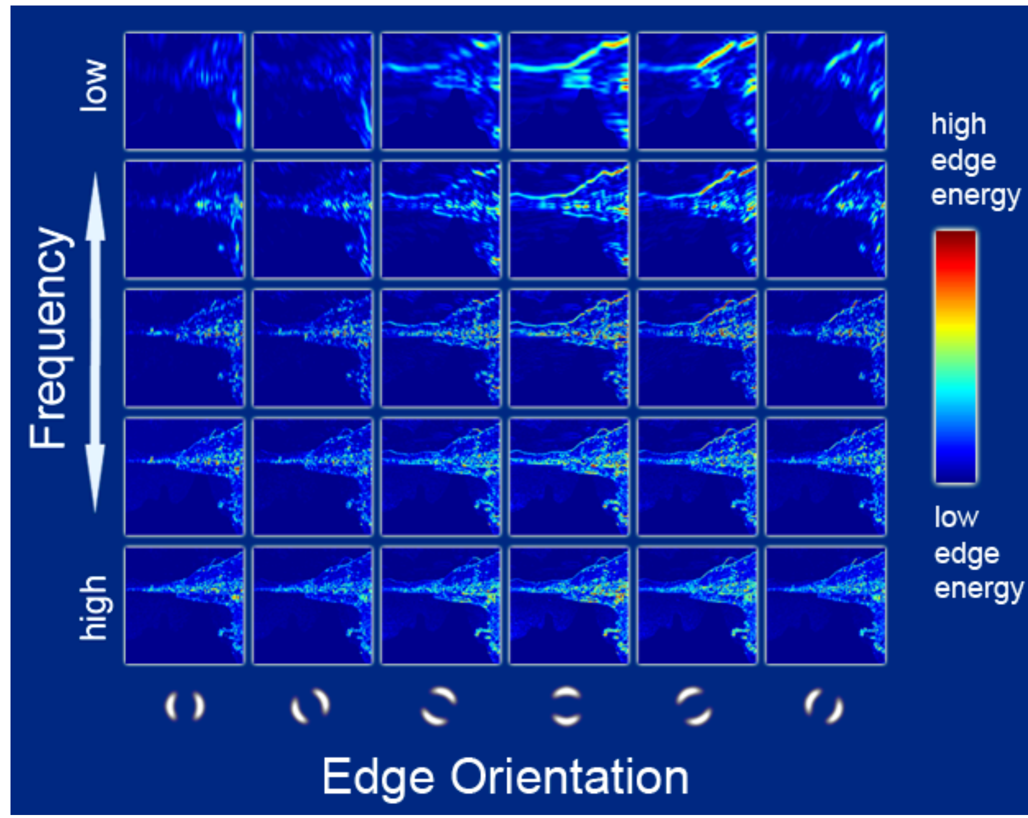
20 completions

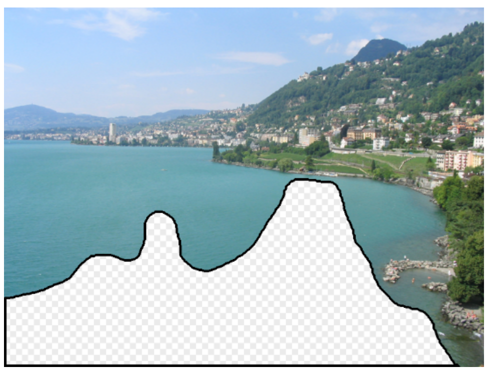
Scene Matching

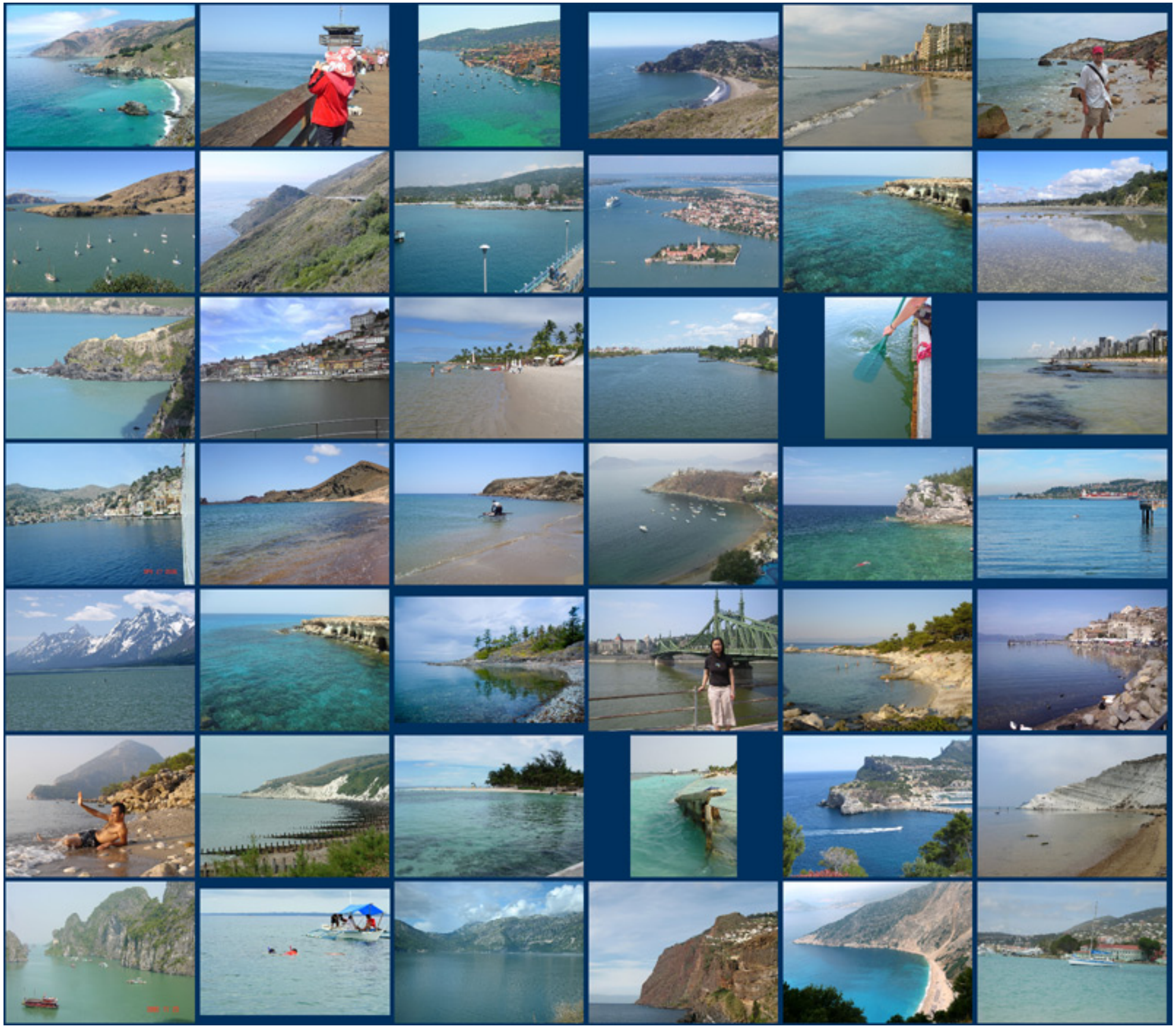


Scene Descriptor



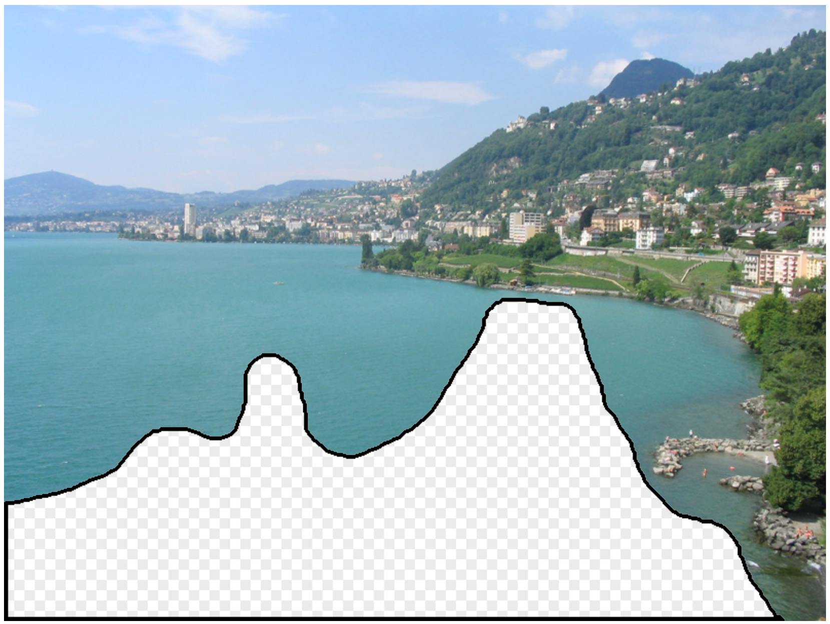




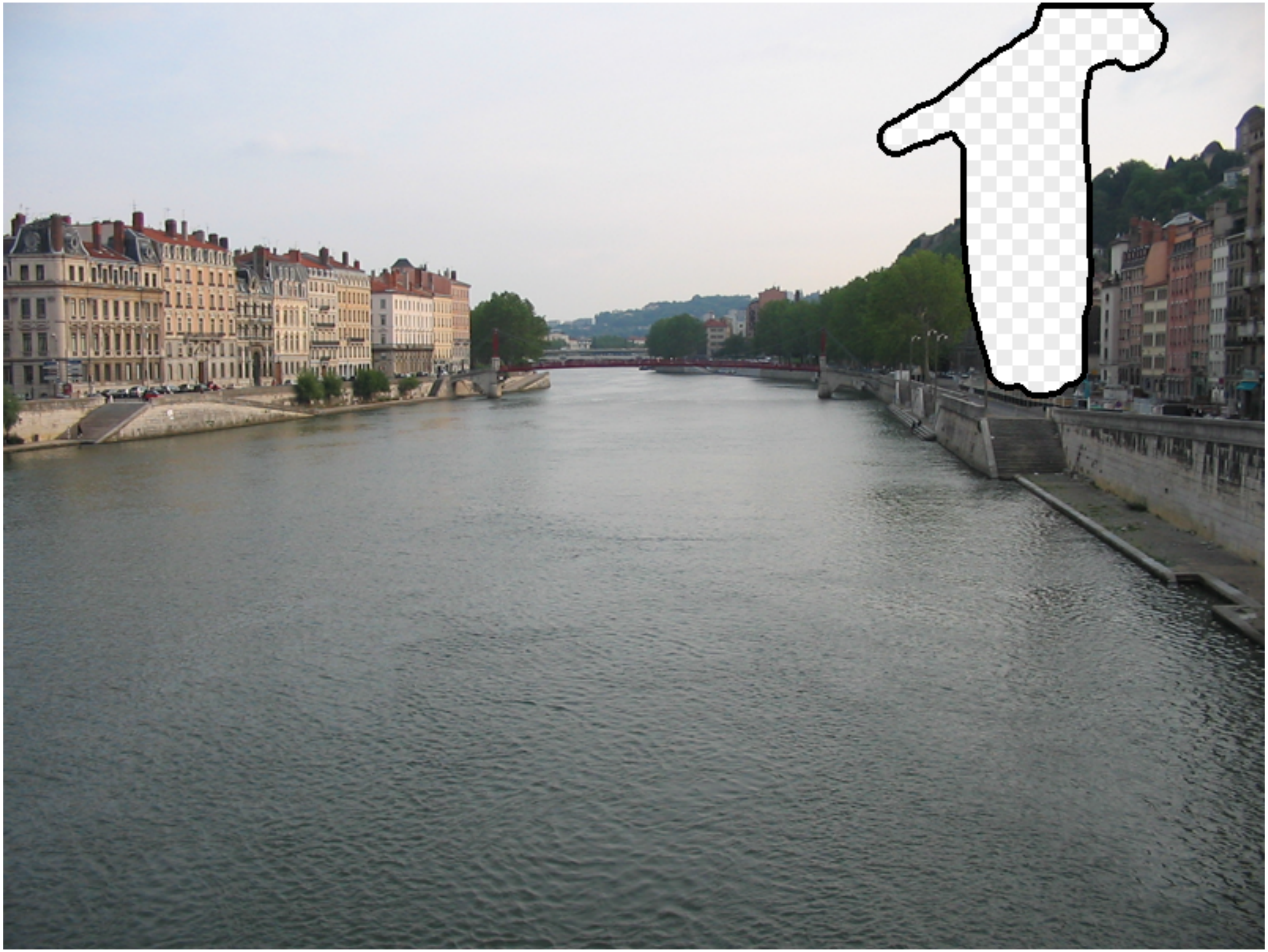


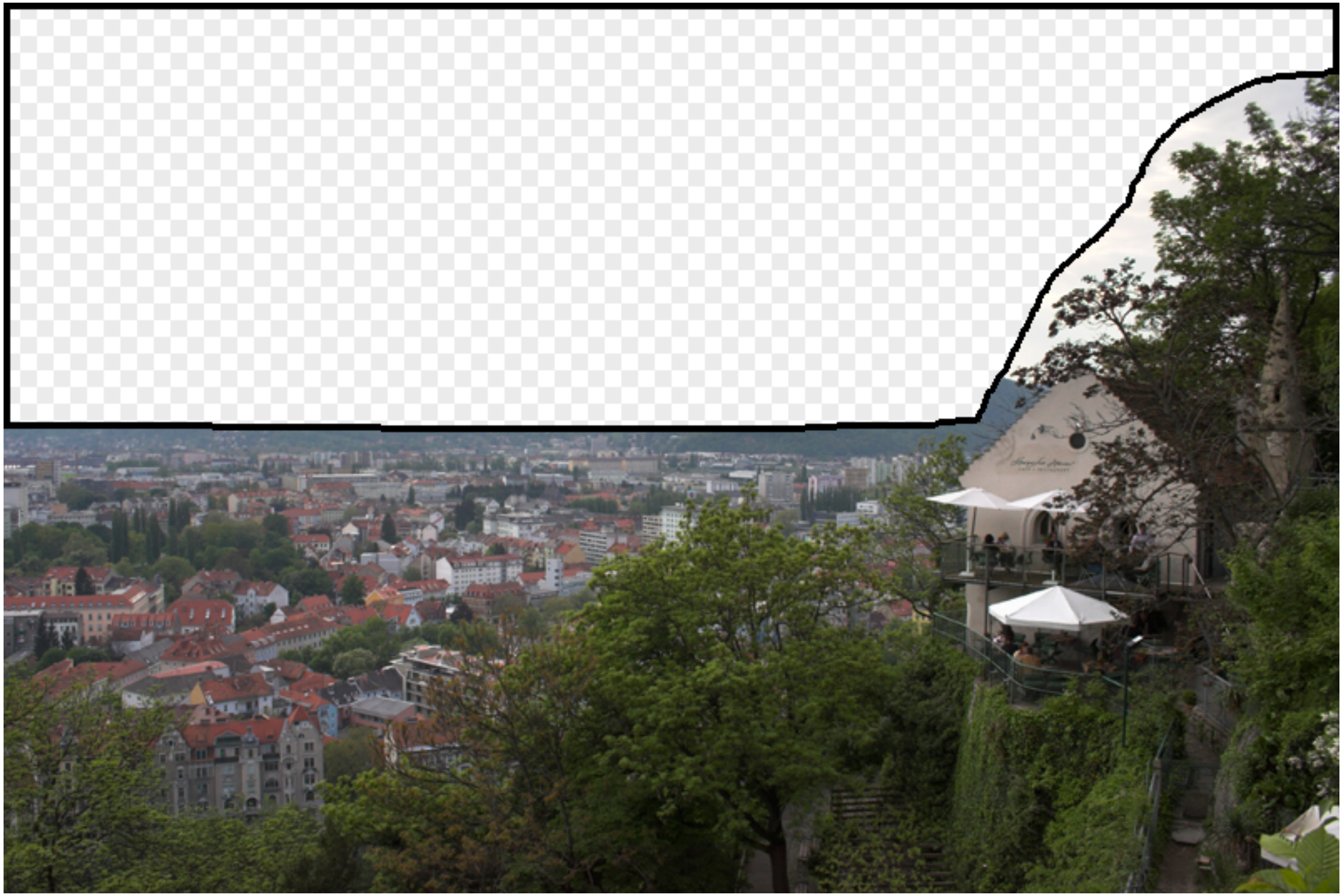
… 200 total

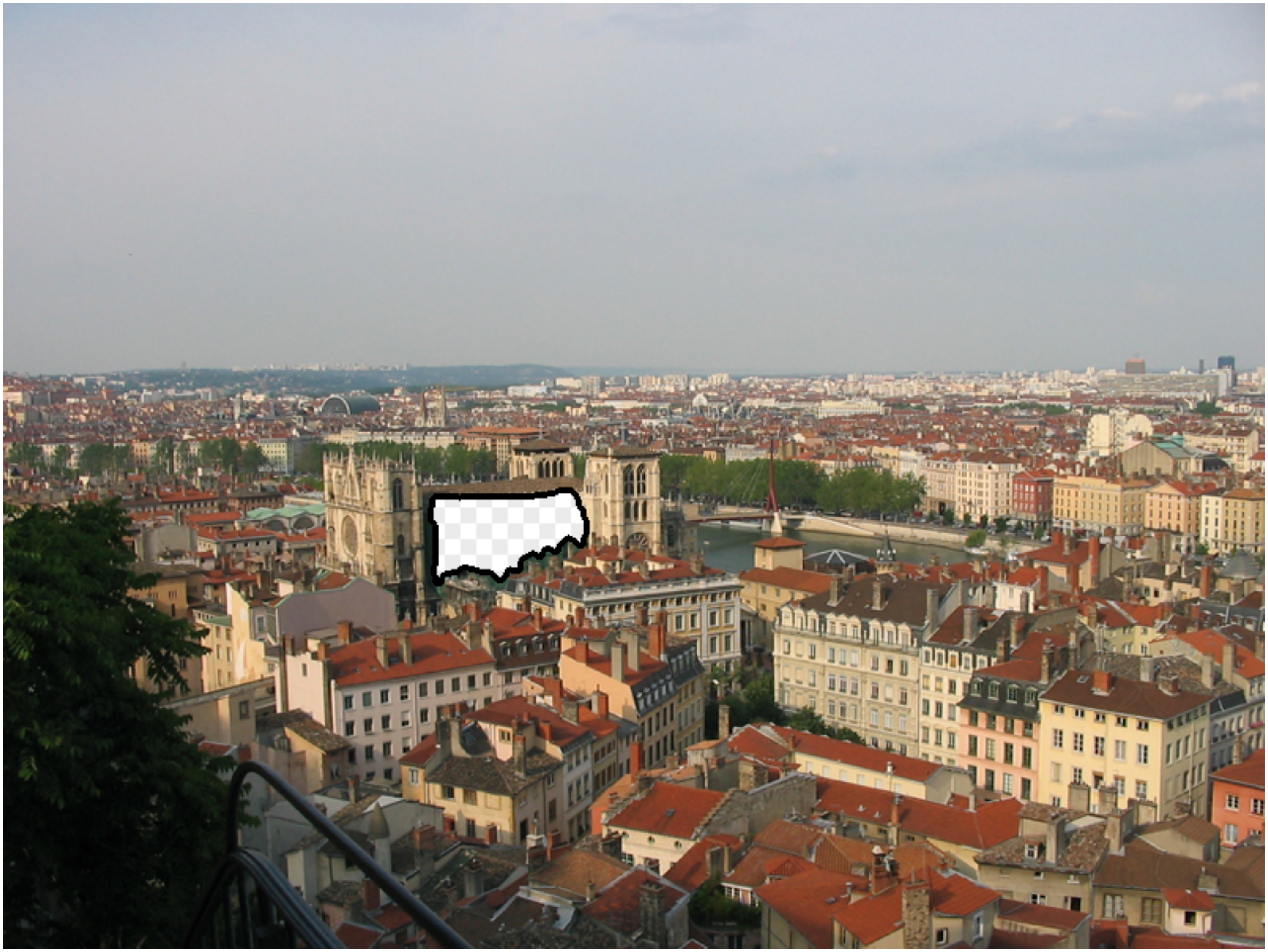
Context Matching




OCT 30 2003







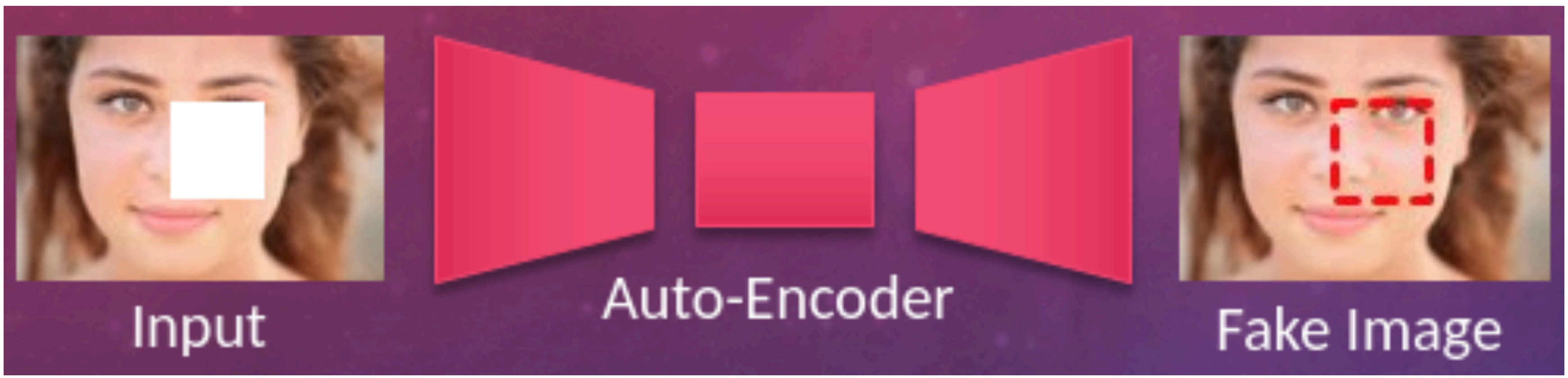
Optimization based super-resolution

Recover high-resolution image by solving an optimization problem

Output high-res image   Input low-res image   Variable

$$I_{out} = \arg\min_{X} \underbrace{\|I_{in} - K \otimes X\|^2}_{\text{Likelihood}} + \underbrace{\lambda \mathcal{R}(X)}_{\text{Regularizer}}$$

Deep learning
– a more powerful data-driven approach



Deep learning based super-resolution

https://bigjpg.com/